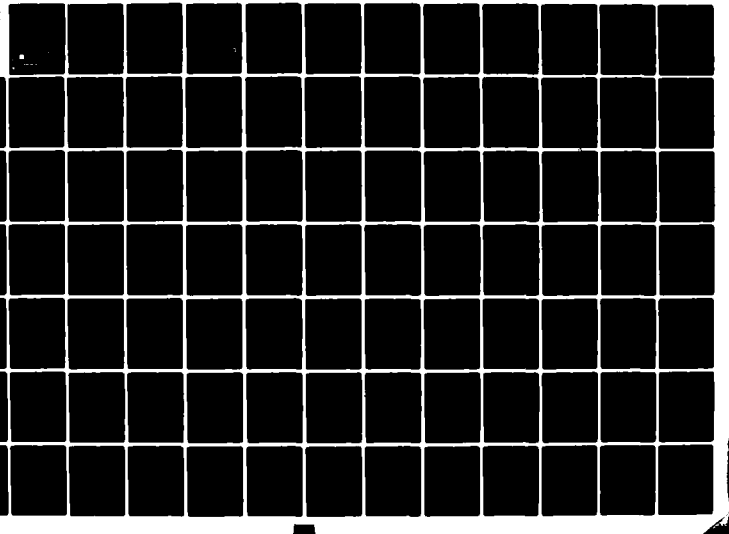
AD-A113 590 CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER --ETC F/G 9/2
A COMPATIBLE HARDWARE/SOFTWARE RELIABILITY PREDICTION MODEL,(U)
JUL 81 X CASTILLO, T D SMITH DAS660-80-C-0057
UNCLASSIFIED CMU-CS-81-138 NL

1 OF 2
AD A
113590



CMU-CS-81-138

(12)

A Compatible Hardware/Software Reliability Prediction Model

Xavier Castillo

22 July 1981

DEPARTMENT
of
COMPUTER SCIENCE

Carnegie-Mellon University

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CMU-CS-81-138	2. GOVT ACCESSION NO. AD-A113 580	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A COMPATIBLE HARDWARE/SOFTWARE RELIABILITY PREDICTION MODEL (U)		5. TYPE OF REPORT & PERIOD COVERED Final Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Xavier Castillo Thomas D. Smith		8. CONTRACT OR GRANT NUMBER(s) DASG60-80-C-0057
9. PERFORMING ORGANIZATION NAME AND ADDRESS Carnegie-Mellon University Department of Computer Science Schenley Park, Pittsburgh, PA 15213		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Ballistic Missile Defense Systems Adv Tech Center ATTN: BMDATC-P/AOLIB P. O. Box 1500, Huntsville, AL 35807		12. REPORT DATE 22 Jul 81 13. NUMBER OF PAGES 148
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE None

16. DISTRIBUTION STATEMENT (of this Report)

"A" Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Submitted to Carnegie-Mellon University in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Design standards
Gaussian distribution (density)
Software(computers)
Weibull distributions (density)
Performance reliability

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(U) In this paper a new modeling methodology to characterize failure processes in Time-Sharing systems due to hardware transients and software errors is presented. The basic assumption made is that the instantaneous failure rate of a system resource can be approximated by a deterministic function of time plus a zero-mean stationary Gaussian process, both depending on the usage of the resource considered. The probability density function of the time to failure obtained under this assumption has a decreasing hazard function, partially

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

BLOCK 20 continued:

explaining why other decreasing hazard function densities such as the Weibull fit experimental data so well. The implications of this methodology are discussed and some applications are given in the areas of Performance/Reliability modeling, software reliability evaluation, models incorporating permanent hardware faults, policy optimization, and design optimization.

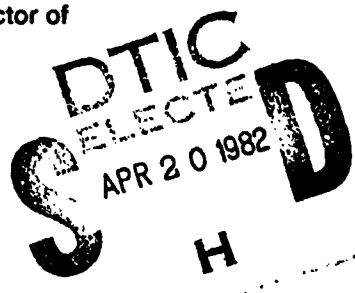
CMU-CS-81-138

A Compatible Hardware/Software Reliability Prediction Model

Xavier Castillo

22 July 1981

Submitted to Carnegie-Mellon University in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering.



This research was supported in part by the Ballistic Missile Defense Systems Command under contract no. DASG60-80-C-0057 and the Fundacion I.T.P., Madrid, Spain.

A la Dolors, amb afecte i admiració,
per la seva comprensió i paciència.

Abstract

In this paper a new modeling methodology to characterize failure processes in Time-Sharing systems due to hardware transients and software errors is presented. The basic assumption made is that the instantaneous failure rate of a system resource can be approximated by a deterministic function of time plus a zero-mean stationary Gaussian process, both depending on the usage of the resource considered. The probability density function of the time to failure obtained under this assumption has a decreasing hazard function, partially explaining why other decreasing hazard function densities such as the Weibull fit experimental data so well. Furthermore, by considering the Operating System kernel as a system resource, this methodology sets the basis for independent methods of evaluating the contribution of software and hardware to system unreliability. The modeling methodology has been validated with the analysis of a real system. The predicted system behavior according to this methodology is compared with the predictions of other models such as the exponential, Weibull, and periodic failure rate. The implications of this methodology are discussed and some applications are given in the areas of Performance/Reliability modeling,software reliability evaluation, models incorporating permanent hardware faults, policy optimization, and design optimization.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my advisor, Dan Siewiorek, and my thesis committee, Steve Director, Bob Fontana, Floyd Humphrey, and Anita Jones, for all the help they gave me in making this thesis reasonably coherent. Mickey Tsao and Steve Elkind also provided valuable critical feedback. Conversations with Marcel Coderch from MIT were specially helpful in formulating certain mathematical problems and their solutions.

The work presented in this thesis critically depends on some measuring tools rarely available in most computing systems. Stephen McConnel, author and maintainer of SEADS, updated the software package according to the needs of the present work. Craig Everhart provided the core of SYSMON, the program with which resource utilization functions have been measured. Howard Wactlar allowed SYSMON to be running permanently on the CMU-10A for several months, even though it meant the permanent loss of one job slot in an already overloaded system. Luis Vidigal wrote the APL program implementing the Powell algorithm used to compute the maximum likelihood values of the parameters of several functions.

Also, I would like to thank my classmates and friends at CMU, who struggled alongside me during these years, especially Mark Carlotto, Andrzej Strojwas and Karem Sakallah. Special thanks to Steve Director for providing a stimulating working environment.

Chapter 1
Introduction

In the early 1950's, the mathematician John Von Newmann studied problems in the design and construction of digital computing machines. In particular, he was interested in the following problem: assume that one has a collection of connected elements computing and transmitting information (an automaton) and each element is subject to eventual malfunction. Can one arrange and organize the elements so that the output is error free for an arbitrary period of time ?

Although encouraging, experience with digital computers in the 1950's had some drawbacks. The ENIAC (Electronic Numerical Integrator and Computer), the first electronic digital computer, had been operating since the mid 1940's. It had 18,000 electronic tubes, each tube having an expected life of 2,500 hours [Goldstine 72]. Von Newmann later proved the feasibility of a computer with 2,500 vacuum tubes and a Mean Time Between Failures of 8 hours, by multiplexing all interconnections 14,000 times, a requirement "not wholly outside the range of our (industrial or natural) experience" [Von Newmann 63].

The fact is that reliability was an overwhelming concern for the designers and users of first generation computers. The components used were relays, vacuum tubes, and delay-line storage devices. All had relatively high failure rates and were subject to transient faults. Hence, fault-tolerant techniques were developed to cope with component unreliability. The use of parity in memories, duplication or triplication and voting, instruction retry, and other hardware fault detection mechanisms were familiar to the designers of those early computers.

The development of fault-tolerance was interrupted by the rather sudden appearance of semiconductor circuits and ferrite cores as digital system components. Hardware suddenly became so "good" that in the 1960's the responsibility of maintaining operation was relegated by default to the system software. A typical example is the MULTICS system of M.I.T. [Corbato 74].

At present, it is clear that fault-tolerant is a desirable attribute of computing systems. The cost

associated with a computer failure in say, a spacecraft, clearly justifies the use of fault-tolerant techniques. But there are many other , more trivial, applications where unreliability is undesirable. In transaction processing systems, airline reservation systems, or even in general purpose computation centers a system failure or "crash" is associated with a user delay to finish one or several tasks. Unreliability of current digital computing systems does not arise from poor quality of the components but from the continuous growth in systems size and complexity. Typical components reliability is measured in failures per million hours. But the number of components used is usually very large and component malfunction (either temporary or permanent) eventually leads to system failures (at least once a day for typical Time Sharing systems). Furthermore, another factor appears now to be specially relevant to system reliability - the correctness of the programs managing the use of system resources, that is, software reliability.

This thesis deals with reliability characterization of digital computing systems. In particular, it is concerned with the behavior that users will observe in their everyday use of Time Sharing systems, and about quantifying the impact of unreliable behavior at the system level. The approach taken has been motivated mainly by the following two facts:

- The desire to formulate a hardware/software reliability prediction model.
- The good fit of the Weibull distribution to experimentally obtained failure data of several computing systems [McConnel 79a].

These two points deserve some more explanation.

1.1. Hardware/Software reliability prediction

The following definition has been drafted by the Software Reliability Committee of the IEEE Reliability Society

> **Definition 1:** A *Compatible Hardware/Software Reliability Prediction Model* is a suitable interpretation of hardware and software mathematical relationships for combined computation so as to make feasible prediction of system reliability [SRC 81].

Prediction models describe the mathematical relationships between certain system parameters. A combined hardware/software prediction model would allow the evaluation of the impact of each cause of unreliability on the observed behavior at the system level. Prediction models can be used to refine a design before actually implementing it, or to optimize the policies regulating the use of systems already operational.

At present, no such modeling methodology is available. As will be described in Chapter 2, current hardware and software modeling efforts are unconnected, preventing the formulation of a unified view of system behavior. Perhaps one of the reasons why more fault tolerance is not found in systems today is due to this lack of cost/benefit analysis techniques.

1.2. The Weibull distribution

If the expectation of having a combined hardware/software reliability prediction model is desirable, the findings about the Weibull distribution are intriguing. After collecting failure data from several systems, a research group at Carnegie-Mellon University reached the following two conclusions :

- Hardware unreliability is mainly due to transients as opposed to permanent faults [Siewiorek 78].
- The Weibull distribution fits experimental data extremely well [McConnel 81].

The Weibull distribution was originally presented by Prof. Weibull in an article dealing with fatigue resistance of steel [Weibull 51]. Prof. Weibull's goal was to find a single distribution of wide applicability that would comprise other distributions as special cases. The Probability Distribution Function (PDF) of the time to failure is given in the case of the Weibull distribution by

$$P(t_f \leq \tau) = 1 - e^{-(\lambda\tau)^{\alpha}} \tag{1.1}$$

Note that for $\alpha = 1$ the Weibull distribution becomes an exponential . For $\alpha = 2$ equation (1.1) becomes the Rayleigh distribution. For $\alpha<1$ equation (1.1) has a decreasing hazard function, a concept which will be formally introduced in Chapter 3 but that can be described intuitively as follows. If h(t) is the hazard function of a system at time t, $h(t)\Delta t$ is the instantaneous probability of observing a system failure on the infinitesimal interval $[t, t + \Delta t)$. For the Weibull distribution,

$$h(t) = \frac{\lambda\alpha}{(\lambda t)^{1-\alpha}} \tag{1.2}$$

which, for $\alpha<1$ is a decreasing function of time. Similarly, for $\alpha>1$, h(t) is an increasing function of t. [McConnel 79b] shows that the Weibull distribution with $\alpha<1$ closely fits the distribution of the time to failure of digital computing systems. Therefore, the instantaneous probability of observing a failure in a computing system decreases as the system is operating.

Figure 1-1 illustrates the behavior of the decreasing hazard function Weibull. The system is started at time t_0 and failures occur at times t_1, t_2,... The system is restarted immediately after each failure.

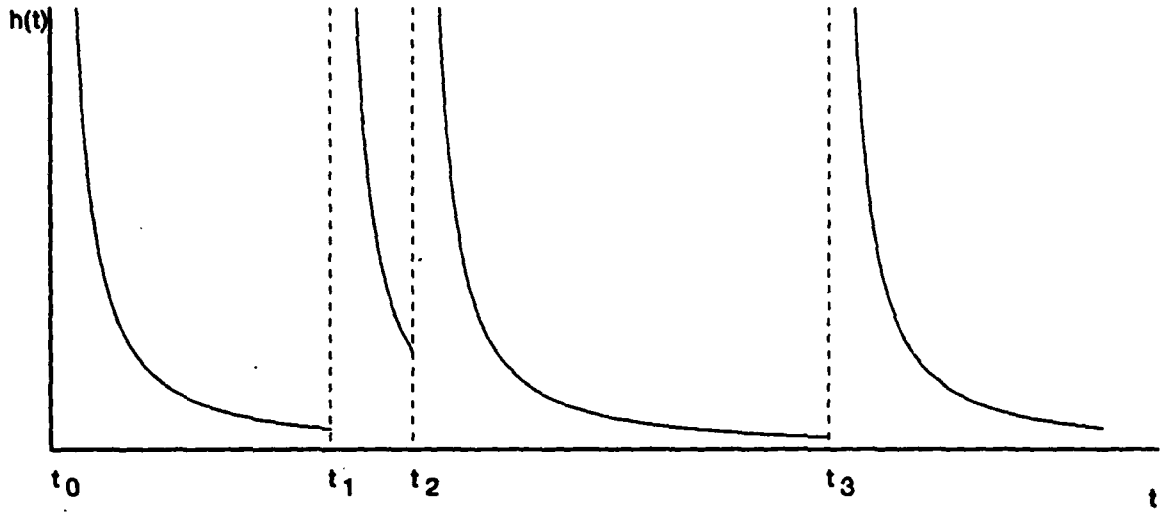


Figure 1-1: Instantaneous probability of failure as a function of operating time for a computing system according to the Weibull distribution

The instantaneous probability of system failure rises after each failure and decreases from then on until the next failure. This behavior is surprising since computers are rarely switched off, therefore not experiencing a warm-up transient period (a possible cause of early failure). Although computer folklore uses the words "cold start" and "warm start" to describe different software initialization sequences, software is also believed to be temperature insensitive. Why should computing systems exhibit such behavior? What are the implications of such behavior? Should all users of a computation center walk out of the terminal room every time that the system is restarted and come back later when the probability of failure is sufficiently small? Or is this just a mathematical paradox irrelevant to system characterization? All these questions will be answered in the following chapters.

1.3. Organization of the research

To quantify the impact of unreliability in a variety of situations, a compatible hardware/software reliability prediction model will be created. This thesis deals with the formulation of such a model, its validation, and the main conclusions draw from its predictions.

Chapter 2 is an overview of current techniques for reliability characterization, causes of unreliability, and existing modeling methods.

In Chapter 3 the proposed modeling methodology is formally developed. The emphasis is on the generality of the results obtained. It is expected that some of these results will be applicable to the characterization of the reliability of other complex systems besides digital computers.

The results presented in Chapter 3 are specialized in Chapter 4, were a study of systems under constant or periodic workload is made.

Chapter 5 discuses the problems associated with model validation. A real system is modeled in detail, and the necessary techniques for measuring and estimating the required model parameters are also given.

Chapter 6 contains a comprehensive study of the similarities and differences between the model presented in this thesis and other modeling efforts. Numerical comparisons between predicted and observed behavior are also given.

Chapter 7 contains some applications derived from the new modeling methodology. Several examples are given which show how to use the model in order to optimize operational policies or quantify the impact of unreliability on the performance of a digital computer system. Although through the Thesis the emphasis is on characterizing unreliability due to hardware transients and incorrect software, an extension of the modeling methods incorporating permanent hardware faults is also given in this Chapter.

Finally, Chapter 8 summarizes the results of the previous chapters and suggests directions for further research.

Chapter 2
Background

The different approaches traditionally used to characterize system reliability will be examined in this chapter. After some definitions given in Section 2.1, the problem of characterizing computing systems reliability is introduced in Section 2.2. Sections 2.3 and 2.4 describe the two major contributions to system unreliability: hardware transients and incorrect software. The modeling methodology presented in this thesis will be validated with a TOPS-10 Operating System. Section 2.5 presents the validation requirements. Amid the description of MULTICS like protection mechanisms (of which TOPS-10 is an example) the new modeling methodology will be presented.

2.1. Definitions

The *reliability* of a system is a measure of how successfully a system conforms to some specification of its behavior. A *failure* is any deviation of system behavior from its specifications. System specifications usually define the *external state* of the system, and failures will be detected as anomalous external states. The following definitions apply only to operational systems, not to systems undergoing development, debugging, or testing (this distinction is important in the case of software reliability modeling). These definitions were first given by [Randell 78].

Definition 1: An *error* is that part of the (internal) system state which is incorrect in the sense that further processing within the specifications of use will lead to a failure.

Definition 2: A *fault* is the electrical, mechanical, or algorithmic cause of an error. A *potential fault* is a fault that *under some circumstances* within the specifications of use will cause an error.

Definition 3: A *permanent hardware fault* is an irreversible electrical or mechanical cause of errors. The internal state of a system in the presence of permanent hardware faults is continuously incorrect.

Definition 4: A *transient hardware fault* is a fault due to temporary environmental, mechanical, or electrical conditions.

Definition 5: A *software fault* is an algorithmic cause of errors.

Note that software and transient hardware faults are always potential faults since they will lead to errors only under certain circumstances of use. For operational systems, transient hardware faults and software faults are indistinguishable in the sense that they are irreproducible errors.

Definition 6: A *system failure* is the external state manifestation of an error such that the entire computing system has to stop operating.

Since no repair takes place after system failures due to software faults or transient hardware faults, the time of system failure is essentially equal to the system restart time. This thesis is solely concerned in modeling hardware transient faults and software faults. Thus, the words *system failure* and *system restart* will be used interchangeably to describe the same event in time.

2.2. The problem of characterizing system reliability

Fault-tolerance has traditionally been characterized by relatively simple functions based on strict assumptions. The *Reliability function* R(t) is defined as the probability of uninterrupted operation up to time t given that all hardware was correctly operating at time t = 0. R(t) may be used to characterize either permanent or transient faults. The usual assumption is made that the failure rate is constant and, for nonredundant systems, the reliability function becomes $e^{-\lambda t}$, where λ is is the sum of the failure rates of all the components in the system. A very common quantitative measure is the Mean Time To Failure (MTTF)

$$MTTF = \int_0^\infty R(t)\,dt \qquad (2.1)$$

The popularity of the MTTF stems mainly from the fact that, for nonredundant systems, it is easily estimated by dividing the time a system is operational by the number of failures reported. Other reliability indices used to compare two systems A and B, are the Reliability Improvement factor (RIF) [Anderson 67]

$$RIF = \frac{1-R_A(t)}{1-R_B(t)} \tag{2.2}$$

and the Mission Time Improvement Factor (MTIF) [Bouricious 69]

$$MTIF = \frac{T_A}{T_B} \quad \text{when } R_A(T_A) = R_B(T_B) = R_{min} \tag{2.3}$$

which are useful only when the system under study must be available for a predetermined period of time T called "mission time".

The concept of *coverage* [Bouricious 69] is defined as the conditional probability of successful recovery, given that a fault has occurred. Although mathematically attractive, coverage has proven to be very difficult to estimate for real systems. Finally, if the Mean Time To Repair (MTTR) is also known, an estimate of the system usefulness given by the *Availability* that for non redundant systems is given by

$$A = \frac{MTTF}{MTTF + MTTR} \tag{2.4}$$

2.2.1. Performance-Reliability evaluation

The above measures do not take into account the performance of the system whose reliability is being measured. Consider Table 2-1 which lists the results obtained from seven different experiments whose goal was explicitly to gain experience on systems reliability. Data for the first system [Yourdon 72], was obtained from a summary of failure statistics on a Borroughs 5500 over a 15 month period starting in April of 1969. Limited information about the cause of each failure is available. For instance, one of the categories includes system failures due to unexpected I/O intercepts. These failures are recorded whenever the software responds to an interrupt signifying that some I/O action has taken place, but discovers that it has no record of having initiated such action. It is thus an indication of some form of hardware or software error but the particular cause for the failure (hardware or

software) remains unknown. The data for the second system was reported in [Lynch 75] and comes from the first thirteen months in the life of an operating system called Chi/OS for the Univac 1108 developed by the Chi Corporation between 1970 and 1973. No explanation is given about how such an accurate decomposition of failures due to hardware and software could be obtained. [Reynolds 75] reports three years of data obtained from a dual IBM 370/165 at Hughes Aircraft Company installed to handle a mixed batch and time sharing load. The fourth system is at the Stanford Linear Accelerator Center (SLAC) where the main workload is processed as multi-stream background batch. The system consists of a foreground host (IBM 370/168) and two background batch servers (IBM 370/168 and IBM 360/91). The architecture is designed to be highly available and reconfigurable. The fifth system is the CMU-10A, an ECL PDP-10 used in the Computer Science Department at Carnegie-Mellon University. The data for the CRAY-1 was reported in [Keller 76], and the data for the three generic UNIVAC systems was reported in [Siewiorek 80].

Table 2-1 gives, when available, a Mean Time to reStart (MTTS) value in hours (that is, the Mean Time to System Failure), a Mean Number of Instructions to Restart (MNIR) which is an estimate of the mean number of instructions executed from system start up until system failure, and the percentages of system failures that were caused by hardware faults, software faults, and faults whose cause could not be resolved. The information about execution rates needed to compute the MNIR value was obtained from [Phister 79].

Obviously, the figures shown in Table 2-1 do not carry much information. A MTTS figure alone does not tell the impact of unreliability on system use. Compare for example the CRAY-1, [Russell 78], with the CMU-10A, [Bell 78]. Although the CRAY-1 crashes twice as often as the CMU-10A, it can operate continuously at rates above 138 Million Instructions Per Second (MIPS), while the CMU-10A operates at 1.2 MIPS. Hence the CMU-10A executes $\sim 10^{10}$ instructions between crashes while the CRAY-1 executes $\sim 10^{12}$ instructions between crashes. Inconsistencies like this one suggest that reliability modeling and measuring should be closely related with the characterization of the performance of the system under study.

Integrated performance-reliability models have already started to appear in the literature. In [Meyer 79], a performance measure called "performability" gives the probability that a system performs at different levels of "accomplishment". In [Gay 79], systems are modeled with Markov processes in order to estimate the probability of being in one of several capacity states. This is a similar approach to the one previously taken in [Beaudry 78], where the concept of "computation reliability" was introduced as a measure which takes into account the computation capacity of a system in each

System	MTTS (hours)	MNIR	% HW	% SW	% Unknown
B 5500	14.7	$2.6\ 10^{10}$	39.3	8.1	52.6
Chi/05 Univac 1108	17	$6.7\ 10^{10}$	45	55	-
dual 370/165	8.86	$2.8\ 10^{11}$	65	32	3
SLAC	20.2	$2.3\ 10^{11}$	73.3	21.6	5.1
CMU-10A	10	$4.3\ 10^{10}$	-	-	-
CRAY-1	4	$1.9\ 10^{12}$	-	-	-
UNIVAC (Large)			51	42	7
UNIVAC (Medium)			57	41	2
UNIVAC (Small)			88	9	3

Table 2-1: Reliability experience of several commercial systems. MTTS is the Mean Time to reStart. MNIR is the Mean Number of Instructions to Restart.

possible operational state. A Performance/Availability model for gracefully degrading systems with critically shared resources is given in [Chou 80]. Finally, in [Moreira 80] a model is described which predicts the cost reduction associated with different values of coverage, repair rate, and diagnosis time. An example shows how the advantage of using a N-redundant system can be quantified assuming only permanent hardware faults.

2.2.2. Causes of unreliability

Most of the above models have been developed mainly for hard failures, that is, stable failures that reflect an irreversible physical change in the hardware. Unfortunately, as it has been repeatedly reported ([Fuller 78], [McConnel 79b], [Morganti 78], [Siewiorek 78], [Ohm 79]), transient failures occur at least an order of magnitude more often than hard failures. A cost effective analysis should then consider transients as the main reason for system unreliability.

Simultaneously with the developments described above, qualitative relationships between workload and unreliability have also been noted. The results published in [Beaudry 79] suggest a strong dependency between workload and reliability of digital computing systems. And in the paper by [Butner 80], this dependency is stated explicitly claiming that a periodic, workload-dependent failure rate is more appropriate to characterize the reliability of time-sharing systems than the classical constant failure rate model traditionally used. As reported in [Castillo 80a], if such a dependency is taken into account it is possible to characterize the performance of digital computing systems considering reliability as an inherent attribute.

Another factor affecting computing systems reliability is the reliability of the software managing the use of system resources. Faults in the software which force a crash and restart operation are not uncommon. They occur, in most commercial systems, much more often than permanent hardware faults, and their effects are similar to the effects of hardware transient faults. The conditions under which a software fault generate an error are usually impossible to determine (as soon as these conditions are determined, the software can be corrected). Hence, the software faults remaining in an operational system are obscure and manifest themselves only upon particular (but unknown) conditions.

In summary, the problems currently relevant to computer reliability characterization are

1. Predominance of hardware transient faults over permanent hardware faults.
2. Software unreliability
3. Disconnection between reliability evaluation and performance evaluation.

The following sections elaborate upon these issues in more detail.

2.3. Hardware transient faults

Hardware transient faults are induced by temporary environmental, electrical, or mechanical conditions. Their effects include flipping a single bit in the main memory of a computer (due to the emission of an alpha particle by radioactive elements present in IC packaging), reading erroneous information from a magnetic disk (due to inaccurate positioning of the reading heads), resetting all CPU registers (due to a power glitch), or receiving erroneous information from a bus (due to electromagnetic radiation received by a bus acting like an antenna).

Although a given transient may occur more than once in the lifetime of a computer, its effects are essentially unpredictable. Consider a single bit in memory that flips its value due to the emission of an alpha particle. If that bit is not storing information at the time that the transient occurs, and the value of the bit is overwritten before being read, the transient passes unnoticed. However, if the same bit is part of a pointer to one of the operating system critical data structures, the entire computer system may crash.

The physical processes generating transient faults generation are presumably sparse since, according to Table 2-1 many commercially available systems are able of continued correct operation for several hours in spite of being built out of a large number of components. Nevertheless, sparseness is not equivalent to total absence and as computing systems become more complex the impact of transient faults may become harder to evaluate unless due attention is paid to this problem during the design process.

2.3.1. Causes of hardware transient faults

Some causes of transients are :

- Limitations in the accuracy of electromechanical devices (such as the positioning servomechanism for the reading heads of a disk drive).
- Electromagnetic radiation received by interconnections (such as long buses acting like receiving antennas).
- Power fluctuations or glitches not properly filtered by the power supply.
- Effects of ionizing radiation on semiconductor devices

This last cause is currently the most important challenge to device designers and requires some

more explanation. It has been only recently that the effects of ionizing radiation have been recognized as a source for "soft" faults in computer memories [May 79]. "Soft" means that the information held in a memory device has changed, but no irreversible change in the device has occurred. Information in computer memories is stored as the presence or absence of charge in capacitors. When an energetic particle creates electron-hole pairs in the vicinity of a capacitor, some of the added charge carriers are collected by the capacitor. If the added charge is sufficiently large, the information stored is changed. The amount of charge that represents a bit and the "critical" charge that is needed to change it have decreased with miniaturization and the advent of VLSI technology. Transient failures in semiconductor memories due to ionizing radiation were not significant until the introduction of 16K bit and 64K bit memory chips.

Two main causes have been detected so far as sources of ionizing radiation which affect the operation of digital computers

- Trace amounts of natural radioactive elements in metallic and ceramic packaging materials [Geilhufe 79]
- The effect of cosmic rays [Ziegler 79]

Although the effect of radioactive materials in packaging materials can be reduced by further purification and better system design, it is not clear how the effects of cosmic rays can be avoided [Keyes 81].

Soft errors are sparse. The designer of a 16 bit 1M word system (built out of 16K dynamic RAM chips) would observe a Mean Time To Soft Failure due to alpha particles of ~40 days [Geilhufe 79]. For a system of the same size (built out of 64K dynamic RAM chips) the Mean Time To Soft Failure due to cosmic rays would be of ~16 days at sea level, ~4 hours at 30,000 feet [Ziegler 79]. However, a soft error is completely removed by the following write cycle. Thus, as pointed out in [Smith 81], the *observed* soft failure rate depends on the frequency between writes or rewrites.

2.3.2. Hardware transient faults modeling

There are three basic approaches to hardware transient modeling. In each approach, the Probability Distribution Function (PDF) of the time to Failure is assumed to be either an Exponential distribution, a Weibull distribution, or an Exponential distribution with periodic failure rate.

2.3.2.1. Exponential distribution

The most widely used model for failure process characterization assumes the failure process to be a homogeneous Poisson process. The PDF of the time to failure is then given by

$$P_e(t<\tau) = 1 - e^{-\lambda_e \tau} \tag{2.5}$$

where λ_e is the (constant) failure rate. The maximum likelihood estimate of λ_e is obtained simply by dividing the time that the system has been operational by the number of failures reported. All functions and parameters related to this model will be noted with subindex "e" and from now on this model will be referred to as the *exponential model.*

2.3.2.2. Weibull distribution

Empirical studies [McConnel 79b] have shown that a Weibull distribution gives a better goodness of fit to experimental data than a simple exponential. The Weibull PDF is given by

$$P_w(t<\tau) = 1 - e^{-(\lambda_w \tau)^{\alpha_w}} \tag{2.6}$$

The Weibull distribution is characterized by two parameters : λ_w , the scale parameter, and α_w , the shape parameter. For $\alpha_w = 1$, the Weibull distribution degenerates to the exponential. For $\alpha_w > 1$, the Weibull distribution has an increasing failure rate. A *decreasing failure rate corresponds to* $\alpha_w < 1$. All reports published to date claim that a *decreasing* failure rate Weibull distribution fits experimental data much better than a simple exponential model. Numerical procedures have been developed to find the maximum likelihood estimates of λ_w and α_w. These procedures are based on the works of [Thoman 69, Berger 74, Romano 77] and FORTRAN programs implementing them are given in [McConnel 79a].

2.3.2.3. Exponential distribution with periodic failure rate

A workload dependent model has been presented in [Butner 80]. A linear or quadratic dependency between failure rate and workload is also assumed. The workload is characterized by a periodic function of time. The proposed PDF becomes an exponential "modulated" by a periodic function

$$P_p(t<\tau) = 1 - e^{-K_p \tau} e^{-F_p U_p(\tau)} \tag{2.7}$$

where F_p is defined as the load induced failure rate, $U_p(\tau)$ denotes the instantaneous load value, and K_p is a workload independent failure rate. This model will be referred to as the *periodic model*, all its parameters having the subindex "p".

Although it has not been used for reliability characterization, a periodic failure rate has been also assumed in a model to determine the optimum checkpointing interval in a transaction processing system by [Chandy 75a]. The assumptions in this model are that transaction processing systems often operate under periodic demand, leading to a periodic failure rate. The optimum checkpointing interval is determined such that the cost associated both with checkpointing and recovering from failures is minimized.

For systems operating under periodic workload an alternative approach is to brake a period into M discrete intervals. The system workload and failure rate are assumed to be constant in each interval. This approach has been used by [Chandy 75b] to evaluate the optimum checkpointing interval in transaction processing systems, and by [Beaudry 80] to characterize the reliability of a multiprocessor for avionics applications and the reliability of the SLAC system described in Section 2.2.1.

2.3.2.4. Discussion

The popularity of the exponential model arises mainly from its simplicity. The exponential model may be a useful abstraction to characterize how failures *occur*. However the validity of the exponential model is not sustained by the data collected about how errors are *detected*. A Weibull distribution with $\alpha_w<1$ seems a much better choice. On the other hand, for systems in steady state operation, the periodic model tries to incorporate the fact that the observed unreliability should depend on patterns of usage, not on a constant set of parameters as the Weibull model implicitly implies. This apparent conflict will be solved in the present thesis, where it is shown that each of the above three models is a special case of a more general characterization.

2.4. Software Reliability

The problem of software reliability assessment is part of the more general area of software quality assessment [Mohanly 73]. Effective mechanisms for measuring software quality are required due to the high cost of software development and maintenance. By 1985 forecasts indicate that over 90% of the total computing dollars spent annually will be for software [Horowitz 75]. The development of techniques for measuring software reliability has been motivated mainly by project managers that require models to estimate the man-power needed to develop a software system with a given level of performance and measuring techniques to detect when this level of performance has been reached. However, most software reliability models presented to date are far from satisfying these two needs in a general context.

There are basicly two approaches dealing with the design of reliable software. The first approach consists in specifying the desired software behavior as accurately as possible and to develop error free software according to the specifications. Thus, this approach deals with the development of fault-intolerant software, and it implicitly assumes that it is possible to develop software packages of arbitrary size and complexity which are completely error free. The second approach acknowledges the fact that to write completely error free software is either impossible or excessively costly. Thus, fault-tolerant software is written which takes into account the possibility of software faults and provides mechanisms for recovering from their effects.

Unfortunately, the words "software reliability model" usually refer to mathematical models dealing with software reliability assessment *during the design of fault-intolerant software*. This is a much more restrictive concept than the general set of tools used to predict, calibrate, and characterize the reliability of software in a variety of environments (which is what the words "software reliability model" would suggest to a novice).

2.4.1. Fault-intolerant software reliability assessment

Software reliability models (in the restricted sense described above) can be roughly grouped into four categories. The first category would include models formulated in the time domain. These models attempt to relate software reliability (characterized, for instance, by a MTTF figure under typical workload conditions) to the number of bugs present in the software at a given time during its development. Typical of this approach are the models presented in [Shooman 73], [Musa 75], and [Jelinsky 73]. Bug removal should increase MTTF and correlation of bug removal history with the time evolution of the MTTF value may allow the prediction of when a given MTTF value will be reached. An example of the application of time domain models to the development of a real-time system is given in [Miyamoto 75]. The main disadvantages of time domain models are that they do not usually take into account that bug correction can generate more bugs, and that software unreliability can be due not only to implementation errors (bugs) but also to design (specification) errors.

Another approach to software reliability modeling is based on studying the data domain. The first model of this kind was described by [Nelson 73]. In principle, if sets of all input data values upon which a computer program can operate are identified, an estimate of the reliability of the program can be obtained by running the program for a subset of input data values. A more detailed description of data domain techniques is given in [Thayer 78]. In the paper by [Schick 78] the time domain and data domain models are compared. However, different applications will tend to use different subsets of all

possible input data values, "seeing" different reliability values for the same software system. This fact is formally take into account in [Cheung 80], where software reliability is estimated from a Markov model whose transition probabilities depend on a user profile. Techniques for evaluating the transition probabilities for a given profile are given in [Cheung 75].

The third category includes models in which software reliability (and software quality in general) is postulated to obey certain laws [Ferdinand 74], [Fitzsimmons 78]. Although such models have generated large amounts of interest, their general validity has never been proven and, at most, they only give a figure for the number of bugs present in a program.

Finally, there have been some attempts to characterize total system reliability (hardware and software) in [Costes 78], and warnings about how not to measure software reliability [Littlewood 79].

What all the above models have in common is that none of them characterizes system behavior accurately enough so as to give the user a figure of guaranteed level of performance under general workload conditions. They concentrate in estimating the number of bugs present in a program but do not give any accurate method to characterize and measure operational system unreliability due to software. There is a wide gap between the variables that can be easily measured in a running system and the number of bugs in its operating system. However, a cost effective analysis should allow the evaluation of software unreliability from variables easily accessible in an operational system, without knowing the details of how the operating system has been written.

2.4.2. Fault-Tolerant Software

Fault-tolerant software assures the reliability of the system by use of protective redundancy at the software level. There are two main strategies for obtaining fault-tolerant software:

- Recovery Blocks
- N-Version programming.

The Recovery Blocks (RV) strategy [Randell 75, Lee 79] consists of three entities: A primary alternate (A_1), an acceptance test (AT), and a list of supplementary alternates (A_2,.....,A_{N-1}). Upon normal execution, A_1 is executed first. If AT is passed, normal computations proceed. If AT is not passed, a purging of data is performed and a new alternate is called. Some modeling efforts for the Recovery Blocks strategy have been reported in [Hecht 76].

The N-Version programming (NV) strategy [Avizienis 75, Avizienis 77] requires N > 1 *independently designed* programs (versions) for the same function. The results after each stage of computation are compared and in the case of disagreement, a preferred result is identified. If redundant hardware is available, the N versions can be executed concurrently. Otherwise, a performance penalty is paid since the N versions have to be executed serially on the same hardware. In [Grnarov 80] the processing times and reliability performance of the RV and NV strategies are compared.

Because of development costs, fault-tolerant software can be found in only a few systems with exceptional reliability requirements, such as space or military systems. Thus fault-tolerant software will not be modeled here since it is not available in the majority of commercial systems.

2.4.3. Discussion

In [Glass 81] a study about "persistent" software errors is summarized. A software error is defined to be persistent if it eludes early detection efforts and does not surface until the software is operational. One of the findings of this study is that a large percentage of persistent software errors are instances of the software not being sufficiently complex to match the problem being solved. It seems as if the programmers were straining to comprehend the complex interrelationships of a problem solution and failed. The analysis section of a Software Problem Report presented by [Glass 81] as a typical example literally describes the cause of a bug as "insufficient brain power applied during design". A large number of errors are the result of a predicate not having enough conditions, or of a variable not being reset to some value after a major piece of code has finished dealing with it.

Unreliability due to software in operational systems is therefore mainly due to persistent errors. That is, the complexity of the data to be processed has been oversimplified in some situations. When one of these situations arise, a software error is generated. Since once it has been written the software does not change, one would be tempted to view the software and all its attributes as static entities. However, this is not what is observed in most operational systems. Although the software is static, the complexity of the data to be processed changes dynamically according to workload and use of the system. Therefore, the view of software reliability as a static property may be useful for software designers, but it is certainly inadequate for users wishing to evaluate the impact of software unreliability in a variety of working environments. The observed software unreliability in an operational system is a dynamic attribute depending (at least) on the following two factors:

- How much the software is used (number of executions per unit time)

- In what way is the software used (what is the complexity of the data to be processed)

These two points will be elaborated in Section 2.5.2.

2.5. Model verification requirements

The previous sections have summarized some current problems associated with reliability characterization of digital computing systems. As stated in Chapter 1, the main goal of this thesis is to provide a modeling methodology able of providing better reliability characterization, particularly relating to the effects of hardware transients and software faults in operational systems. The following constraints have also been imposed:

- The characterization should be user oriented. That is, it should provide users with a set of tools to evaluate the impact of unreliability due to software and hardware transients. This contrasts sharply with most software reliability models, which are oriented to help the designer to meet a static requirement.
- No matter how complex the model may be, the results should be easy to understand and apply.
- Model parametrization must be possible from easily measurable variables in operational systems. Situations such as the ones created after the introduction of "coverage" (conceptually attractive, but impractical to measure in real systems) should be avoided.
- The model must be validated by contrasting its predictions with the behavior of real systems.

The last restriction is particularly important since validation will be possible only with systems which have the necessary measuring tools already incorporated. Because of its availability at CMU, validation will be made with the TOPS-10 Operating System, a MULTICS like Time Sharing system. Since most commercially available Time-Sharing systems have protection mechanisms based on the original MULTICS design, this is not a particularly restrictive constraint. However, MULTICS protection mechanisms themselves may give some hints about how the analysis should be started.

2.5.1. The MULTICS Time Sharing system

MULTICS (MULTiplexed Information and Computing Service) [Organick 72] was designed in the mid 1960's as a prototype of a computer utility. Among other goals, it was to provide convenient remote terminal access, continuous operation analogous to that of electric power or telephone companies, and the ability to support different programming environments. Thus, one of the requirements was to provide facilities for the protection of concurrently executing programs. The protection mechanism proposed for MULTICS (and originally implemented in software) was named *rings of protection.* Conceptually, an executing program segment in MULTICS is executing in one of a set of concentric rings. A program can access programs and data in the rings outer to its ring. But data in inner rings is only accessed through predefined "gates". By subsetting the segments of a process into rings and by effectively controlling interactions and communication between segments in different rings, MULTICS provides the potential to isolate trouble and limit damage. Different rings are equated to different levels of damage.

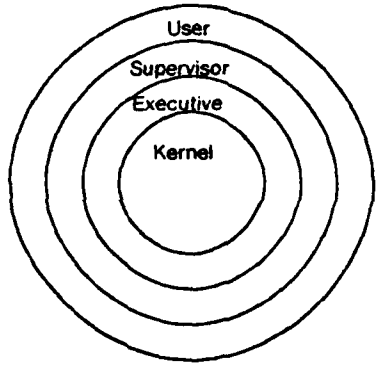


Figure 2-1: Rings of protection in a typical Time Sharing system

Later systems have typically four rings of protection (Figure 2-1) and have the necessary hardware mechanisms to enforce protection across them. The innermost ring or *kernel* is the most privileged and the closest to the hardware. I/O interrupt routines, schedulers, pagers, and the most critical operating system data structures reside in the kernel. The outer rings have different levels of privilege and responsibility. In a typical partition, I/O formatting and operating system services are executed in the next ring or *executive*, command parsing and real-time jobs execute in the following ring or *supervisor* and the last ring (the least privileged) is reserved for the execution of user processes and run-time libraries.

2.5.2. A starting point

One of the nice properties of having rings of protection is that software singularities are readily identified. Assuming perfect recovery from faults in the outer rings, the entire system collapses only when the kernel software cannot execute. But the kernel software will always execute properly unless

- Hardware transient errors corrupt the kernel code or data structures
- The kernel software itself contains faults that under certain conditions corrupt itself or its data structures
- Some hardware does not exist because of a permanent hardware fault

The *observed* reliability due to hardware transients and software faults will therefore depend on how much the kernel is used. Given that transients occur at random, the longer the systems executes in kernel mode, the more likely is that a transient will affect the kernel software or its data structures. Analogously, the probability that a software fault will manifest itself as an error will increase as the kernel is exercised more and more. Thus, the observed unreliability due to hardware transient faults and software faults should be a function of kernel usage.

The assumption of having perfect recovery from faults in the outer rings is too strong and can be partially relaxed. In fact, the two main assumptions on which the thesis is based are:

- Software faults in the kernel are *more likely* to lead to a system failure than software faults in any of the outer rings
- A transient affecting the operation of the kernel software is *more likely* to lead to a system failure than a transients affecting other software.

These two assumptions are compatible with the presence of software faults in outer rings which may abort single jobs, or even occasionally crash the system. The assumptions refer to the *average* behavior of a system in steady state operation and do not negate the possibility of pathological situations (such as the possibility of having a software fault that crashes the system in an almost zero load situation). These two assumptions only suppose that such pathological cases are rare.

In Chapters 3 to 7, the consequences of the above two assumptions will be rigorously formulated, validated, and an investigation of their main implications will be made. At the end of Chapter 7, the modeling methods derived from these two assumptions will be combined with traditional modeling tools and the possibility of permanent hardware failures will be also taken into account. Thus, a combined model taking into account the effects of transient hardware failures, software failures, *and* permanent hardware faults will be introduced.

2.6. Summary

This chapter has summarized some of the problems associated with the characterization of computing systems reliability. In particular, it has been shown how independent reliability evaluation and performance evaluation is itself a problem. The main causes of system unreliability (hardware transients and software faults) have also been described, along with current modeling efforts.

Modeling methodologies for hardware transient faults and software faults are completely independent, probably because these modeling methodologies are a response to designer's needs, and component designers rarely interact with software designers.

The approach adopted in this thesis to characterize reliability at the system level is to put more emphasis on *what will be observed* while paying less attention to how and why a given error occurs. The main implicit assumption throughout the thesis is that reliability of complex systems can be characterized by examining the patterns of usage of system singularities. The more a singularity is used, the more likely it is that a failure will be observed.

For (ideal) Time Sharing systems, the main singularity is the kernel of the operating system. The kernel can be damaged either because of transients or because of kernel software errors. The more the kernel is exercised, the more likely that a transient will affect its operation or that a software fault will generate an error. The formal framework for this approach is presented in the following chapter.

Chapter 3
Mathematical formulation

This chapter gives the mathematical basis of a model capable of predicting the unreliability of digital computers due to hardware transients and software faults. The results are essentially theoretical and will be validated by means of analyzing real systems behavior in subsequent chapters. Although the main goal is to develop a mathematical framework suitable to the characterization of the reliability of MULTICS like Time Sharing systems, the results obtained in this chapter are expected to apply to a wider class of complex systems, namely, those systems with a failure rate that can be approximated by either a stationary or cyclostationary Gaussian process. All the approximations and specializations to computing systems analysis will be worked out in Chapter 4. The results presented in this Chapter are closer to applied probability theory than to computing systems characterization. For the reader not interested in strictly mathematical results the introduction to Section 3.2, Section 3.2.1, and the summary at the end of the chapter should be enough to give an idea of the main results.

In Section 3.1 the necessary definitions are given and the notation used through the thesis is introduced. Section 3.2 is devoted to the description of the process underlying the unreliable behavior of digital computer systems. The emphasis is not on why and how often faults are generated, but on what the system is doing when an error is detected. The reliability of the system is shown to depend on an integral converging to a Gaussian random variable and, more generally, to a Wiener process. However, its evaluation requires some statistics which are impractical, if not impossible, to evaluate from real systems. Thus, in Section 3.3 an approximation is given which depends only on easily measurable variables. The Probability Distribution Function of the time to failure is shown to be completely characterized by a single time function in Section 3.4, leading to a conceptually equivalent but simpler description of the failure process than the description based on convergence concepts. Finally, a summary of the results presented in this Chapter is given in Section 3.7.

3.1. Definitions and notation

A *Probability Space* $(\Omega,\mathcal{A},\mathcal{P})$ is comprised of a sample space Ω, a collection of subsets of Ω forming a sigma field (written σ-field) of events, $\mathcal{A}$, and a probability measure $\mathcal{P}$. An element $\omega\in\Omega$ is a possible outcome of a random experiment. A subset of Ω (a collection of possible outcomes) is called an event. In general, not all collections of outcomes are observable events. Probability theory deals only with events in a σ-field. A σ-field of sets is a collection of sets closed under complementation, union, and countable unions. The reason for associating observable events with a σ-field is that whenever we have a sequence of observable events $\{A_i\}$, the fact that their complements $\{A_i^C\}$ and countable union $\bigcup_{i=1}^{\infty}A_i$ are also observable events facilitates the proofs of many basic probability theory results. Finally, the probability measure $\mathcal{P}$ is a function that maps each set in $\mathcal{A}$ into the unit interval [0,1].

Definition 1: A *random variable* $x(\omega)$ is a function with domain Ω and range the real line $\mathbb{R}$ such that for every Borel set X in $\mathbb{R}$, the set $\{\omega|x(\omega)\in X\}$ is in the σ-field of events $\mathcal{A}$.

The definition ensures that the probability of any event of the form $P(\{\omega|x(\omega)\in X\})$ is well defined for any subset X of $\mathcal{B}$, where the Borel sets $\mathcal{B}$ are the subsets of $\mathbb{R}$ belonging to the smallest σ-field generated by the set of all closed intervals. Sometimes it will be necessary to refer to all possible events associated with a random variable or with a collection of random variables $x_1,\ldots,x_k$. Such collection of events will be a σ-field and will be denoted by $\sigma(x_1,\ldots,x_k)$ meaning the smallest σ-field containing all the sets of the form

$$\{\omega|x_1(\omega)\in X_1,\ldots,x_k(\omega)\in X_k\} \qquad X_1,\ldots,X_k\in\mathcal{B}$$

Definition 2: The *Probability Distribution Function* (PDF) of a random variable x is defined as the function

$$P_x(x\leq\xi) = P(\{\omega|x(\omega)\leq\xi\})$$

The PDF of a random variable maps the real line $\mathbb{R}$ into the unit interval. It is a nondecreasing function of ξ and $P_x(x\leq-\infty) = 0$, $P_x(x\leq\infty) = 1$. If there exists a nonnegative function $p_x(u)$ such that

$$P(x\leq\alpha) = \int_{-\infty}^{\alpha} p_x(u)\,du$$

the p_x is said to be the *probability density function* (pdf) of x.

Of particular importance is the *Gaussian* or *Normal* distribution. If a random variable x is Gaussian (or normally) distributed, then

$$P(x \leq \alpha) = \frac{1}{(2\pi)^{1/2}\sigma} \int_{-\infty}^{\alpha} e^{\frac{(u-m)^2}{2\sigma^2}} du \tag{3.1}$$

where m and σ^2 are respectively the expected value and variance of x. The normalized Gaussian distribution (with zero mean and variance 1) will be noted $\Phi(\alpha)$

$$\Phi(\alpha) = \frac{1}{(2\pi)^{1/2}} \int_{-\infty}^{\alpha} e^{-u^2/2} du \tag{3.2}$$

Definition 3: A stochastic process $\{x_t(\omega); t\in T, \omega\in\Omega\}$ is a family of random variables all defined in the same probability space Ω and indexed by a real parameter t that takes values in a parameter set T called the *index set* of the process.

The indexing parameter t will represent time in all the processes presented in this thesis and T will always be equal to the real line $\mathbb{R}$, that is, only continuous time processes will be considered. For each fixed $t\in\mathbb{R}$, $x_t(\omega)$ as a function of ω will be a real valued random variable. For each $\omega\in\Omega$, $x_t(\omega)$ as a function of t will be a real valued function of time called a *realization* or *sample function* of the process. The set of all these time functions is called the *ensemble* of the process. A *sequence* (or a countable stochastic process) of random variables $x_1(\omega)$, $x_2(\omega)$,... is a particular form of a stochastic process in which the index set is the nonnegative integers $\mathbb{N}^+$.

Stochastic processes will always be denoted such that time dependency will be expressed as a subscript, while deterministic functions of time will have the argument in parenthesis. Thus, x_t is a stochastic process and h(t) is a deterministic function of t.

The following convergence concepts will be needed later in the chapter.

Definition 4: Convergence in probability. The sequence $\{x_i(\omega)\}$ is said to *converge in probability* to $x(\omega)$ if for every $\epsilon > 0$

$$\lim_{n\to\infty} P(|x_n - x| > \epsilon) = 0$$

and will be noted as

$$\text{p.lim}_{n\to\infty} x_n = x$$

Definition 5: Convergence in distribution. The sequence $x_1, x_2,...$ is said to *converge in distribution* to x if

$$\lim_{n\to\infty} P_{x_n}(x_n \leq \xi) = P_x(x \leq \xi)$$

and will be noted as

$$P_{x_n}(\xi) \Rightarrow P_x(\xi)$$

Convergence in distribution is weaker as it is implied by convergence in probability.

Definition 6: A *ccunting process* $\{N_t(\omega); t \geq t_0\}$ is a stochastic process having the set $\mathbb{N}^+ = \{0,1,2,..,\infty\}$ of nonnegative integers as its state space.

For each $\omega \in \Omega$, $N_t(\omega)$ is a piecewise-constant function of t with jumps at $t_1(\omega)$, $t_2(\omega)$,...,$t_n(\omega)$, the values of t_1 ,...., t_n *depending on the realization of the process.* Counting processes are always associated with *point processes*, the value of $N_t(\omega)$ for $t_i \leq t < t_{i+1}$ being the total number of "points" generated up to t_{i+1}. All counting processes presented in this thesis will be associated with *failure processes* of a given system, the value of $N_t(\omega)$ for $t_i \leq t < t_{i+1}$ being the number of failures detected up to t_{i+1}.

Definition 7: A *renewal process* is a counting process where the time durations between consecutive events are positive, independent, identically distributed random variables.

Renewal processes are commonly used for reliability modeling. In the case of permanent hardware faults, it is assumed that after repair has been done the system is as good as new. Thus, the times between successive permanent hardware faults verify the conditions given in the above definition and the failure process is usually assumed to be a renewal process.

Definition 8: A *Poisson process* is a counting process $\{N_t ; t \leq t_0\}$ with the following three properties :

1. $Pr[N_{t_0} = 0] = 1$
2. For $t_0 \leq s < t$, the increment $N_{s,t} = N_t - N_s$ is Poisson distributed with parameter $\Lambda_t - \Lambda_s$, where Λ_t is a nonnegative, nondecreasing function of t.
3. $\{N_t; t \geq t_0\}$ has independent increments.

Property 3 is the distinguishing property. It means that for a Poisson counting process, the number of points in nonoverlapping intervals are statistically independent random variables, no matter how large or small the intervals are and no matter how distant or close they may be. The function Λ_t in property 2 is termed the *parameter function* of the process. If Λ_t is an absolutely continuous function of t, it can be expressed as

$$\Lambda_t = \int_{t_0}^{t} \lambda_\tau \, d\tau \qquad (3.3)$$

where λ_τ is a nonnegative function of t for $t \geq t_0$. The function λ_τ is termed the *intensity function* of the process N_t. At any time $t \geq t_0$, the intensity function $\lambda(\tau)$ is the instantaneous average rate at which points occur. If N_t is a failure process λ_t is the *failure rate* of the process.

Definition 9: A Poisson process is said to be *homogeneous* when the intensity function λ_t is a constant λ independent of time.

For an homogeneous Poisson process, the PDF of the time to the next failure t_f given that the system is observed since time t_s is given by the Exponential distribution

$$P(t_f \leq \tau | t_s) = 1 - e^{-\lambda(\tau - t_s)} \tag{3.4}$$

where λ is the mean rate at which points (failures) are generated.

Definition 10: Whenever the intensity function λ_t is not a constant but a deterministic function of time $\lambda(t)$, the corresponding Poisson process is said to be *nonhomogeneous*.

For a nonhomogeneous Poisson process, the PDF of the time to the first failure is given by

$$P(t_f \leq \tau | t_0) = 1 - e^{-\int_{t_0}^{\tau} h(t)\,dt} \tag{3.5}$$

where h(t) is termed the *hazard function* of the process. Note that by property 2 in the definition of a Poisson process, $h(t)\Delta t$ is the probability of observing a failure in the infinitesimal interval $[t, t + \Delta t)$. Thus, for a nonhomogeneous Poisson process, the probability of observing a failure in different infinitesimal intervals evolves as a deterministic function of time.

Definition 11: Let x_t be a stochastic process that is an "outside" process influencing the evolution of a counting process $\{N_t; t \geq t_0\}$. N_t is a *doubly stochastic Poisson process* with intensity process $\{\lambda_t(x_t); t \geq t_0\}$ if for almost every realization of the process x_t, N_t is a Poisson process with intensity process function $\lambda_t(x_t)$.

The process x_t carries the information about how the intensity process varies, and for this reason is sometimes called the *information process*.

Definition 12: A *stationary process* (in the strict sense) is a stochastic process $\{x_t\}$ with the property that for any positive integer k and any points $t_1, \ldots, t_k$ and h in T, the joint distribution of

$$\{x_{t_1}, \ldots, x_{t_k}\}$$

is the same as the joint distribution of

$$\{x_{t_1 + h}, \ldots, x_{t_k + k}\}$$

Intuitively, a process is stationary if it has the same joint statistics regardless of where the time origin is set. Hence, if x_t is a *stationary Gaussian process*, the joint distribution function of $\{x_{t_1+h},\ldots,x_{t_k+h}\}$ is a multivariate Gaussian distribution whose covariance matrix is independent of h.

Definition 13: The *Autocorrelation function* $R_{xx}(t_1,t_2)$ of a process x_t is defined as

$$R_{xx}(t_1,t_2) = E[x_{t_1} x_{t_2}]$$

where E[..] stands for expected value.

If x_t is stationary and real, $R_{xx}(t_1,t_2)$ depends only on the time difference $\tau = |t_1 - t_2|$ and

$$R_{xx}(\tau) = E\{x_{t+\tau}x_t\}$$

Definition 14: A stationary Gaussian process will be termed *white noise* if its Autocorrelation function is given by

$$R_{xx}(\tau) = \frac{W}{2}\ \delta(\tau) \tag{3.6}$$

where $\delta(x)$ is the Dirac delta function.

As will be discussed in the following sections, the main difference between white noise and any other stationary Gaussian process is that of *predictability*. While a maximum likelihood estimate of future values of a process exists for nonwhite noise processes, white noise is essentially unpredictable.

Definition 15: A stochastic process $x(t,\omega)$ is *ergodic* in the most general sense if all of its statistics can be determined from a single realization $x(t,\omega_0)$ of the process.

Loosely speaking, a process is ergodic if time averages (the only ones that can be obtained from a single realization of the process) equal ensemble averages (i.e. expected values). Obviously, ergodicity can be defined with respect to certain parameters of the process. Only ergodicity with respect to the autocorrelation function will be needed in this thesis, which is defined as follows :

Definition 16: A stochastic function is ergodic with respect to the autocorrelation function if

$$R_{xx}(\tau) = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x_{t+\tau}x_t\, dt \tag{3.7}$$

with probability one.

If ergodicity of the autocorrelation function is satisfied, the autocorrelation function can be estimated by computing the above integral for a finite record of a single realization of the process x_t.

Definition 17: A real valued, continuous time stochastic process is defined to be a *cyclostationary process* with period T if and only if [Gardner 75]

1. $E\{x_t\} = E\{x_{t+T}\}$

2. $E\{x_t x_s\} = E\{x_{t+T} x_{s+T}\} \quad \forall\, s,t$

that is, it is a stochastic process with periodic mean and autocorrelation functions.

Definition 18: A doubly stochastic Poisson process will be said to be a *cyclostationary Poisson process* if its information process is cyclostationary.

Definition 19: A *Wiener process* is a stochastic process $\{W_t | t \geq t_0\}$ such that $W_{t_0} = 0$ and the joint distribution of

$$(W_{t_n}, \ldots\ldots, W_{t_0}) \quad t_n > t_{n-1} > \ldots > t_0 \geq 0$$

is specified by the requirement that the random variables $x_k = W_{t_k} - W_{t_{k-1}}$, $k = 1,\ldots,n$, be independent, normally distributed random variables with

$$E[W_{t_k} - W_{t_{k-1}}] = \mu(t_k - t_{k-1})$$

$$Var[W_{t_k} - W_{t_{k-1}}] = \sigma^2(t_k - t_{k-1})$$

In particular, note that for fixed t, W_t is a normally distributed random variable with $E[W_t] = \mu t$ and $Var[W_t] = \sigma^2 t$. The Wiener process is an interesting abstraction useful in describing certain physical phenomenon such as the Brownian motion of a particle in a fluid. It has curious mathematical properties such as the fact that although almost all sample functions are continuous, they are nowhere differentiable. However, although being nowhere differentiable, if w_t is white noise,

$$W_t = \int_{t_0}^{t} w_\tau \, d\tau \tag{3.8}$$

in the sense that the integral on the right side of the above equation has all the formal attributes of a Wiener process.

Let C denote the space of all real valued, continuous functions on $[0,\infty)$ and let $\mathcal{C}$ denote the smallest σ-field of C where W_t is measurable. It can be shown that there exists a unique probability measure $\mathcal{W}$ such that $\{W_t, t \leq t < \infty\}$ is a Wiener process. By definition, $\mathcal{W}$ is such that W_t is a Gaussian distributed random variable and $\mathcal{W}(t,\alpha)$ will be used to note

$$\mathcal{W}(t,\alpha) = P(W_t \leq \alpha)$$

$$= \frac{1}{(2\pi t)^{1/2}} \int_{-\infty}^{\alpha} e^{-x^2/2t} \, dx \tag{3.9}$$

3.2. The underlying failure process

The mathematical problem to be solved is summarized in Figure 3-1. As it has been explained in Chapter 2, the problem is to characterize the unreliability of a MULTICS like Time Sharing System due to hardware transient faults and software faults.

Figure 3-1 shows a situation in which hardware transient faults occur in different components of different subsystems (memory, CPU, bus) at times t_1, t_2,..... The sensitivity of the system to the presence of a hardware transient fault depends on what the system is doing at the moment that the transient occurs. If a transient occurs while the system operates in kernel mode the system will crash with probability p_k. If the system operates in user mode at the moment that the transient occurs, the probability of a crash is p_u. It is assumed that $p_k > p_u$. The system may also crash while in kernel mode due to the manifestation of a kernel software fault. The probability of such an event during time interval Δt is assumed to be $p_s \Delta t$ (the assumption that p_s is constant will be relaxed in the following Chapter).

The probability of observing an error in a single component is extremely small, and the number of components, very large. The average Mean Time To Failure (MTTF) of a single component is on the order of 10^6 hours ($\sim 10^3$ years) for hard failures [Hodges 77]. The number of components varies from 10^3 for a small minicomputer like the PDP-11/40 [Bell 78] to 10^5 IC packages for a supercomputer like the CRAY-1 [Russell 78]. Hence the failure process due to transients is equal to the superposition of a large number of very sparse failure processes. It is proved in [Cinlar 72] that this type of superposition converges to a Poisson process. Thus, the system failure process can be viewed as a Poisson process with intensity

$$\lambda_t = \begin{cases} p_k + p_s & \text{if the system operates in kernel mode at time } t \\ p_u & \text{otherwise} \end{cases} \qquad (3.10)$$

λ_t will be termed *failure rate* because it is the rate at which errors leading to a *system failure* are generated.

3.2.1. The underlying intensity process

Let $N_{[t_1,t_2]}$ be the counting process which counts the number of system failures in the interval $[t_1,t_2]$. Whether the system operates in kernel or user mode depends on user requests for program execution and on program behavior. But it is certain that requests to the kernel will arrive at random and that the duration needed by the kernel to satisfy each request will be also random. λ_t is therefore a stochastic process and $N_{[t_1,t_2]}$ becomes a doubly stochastic Poisson process.

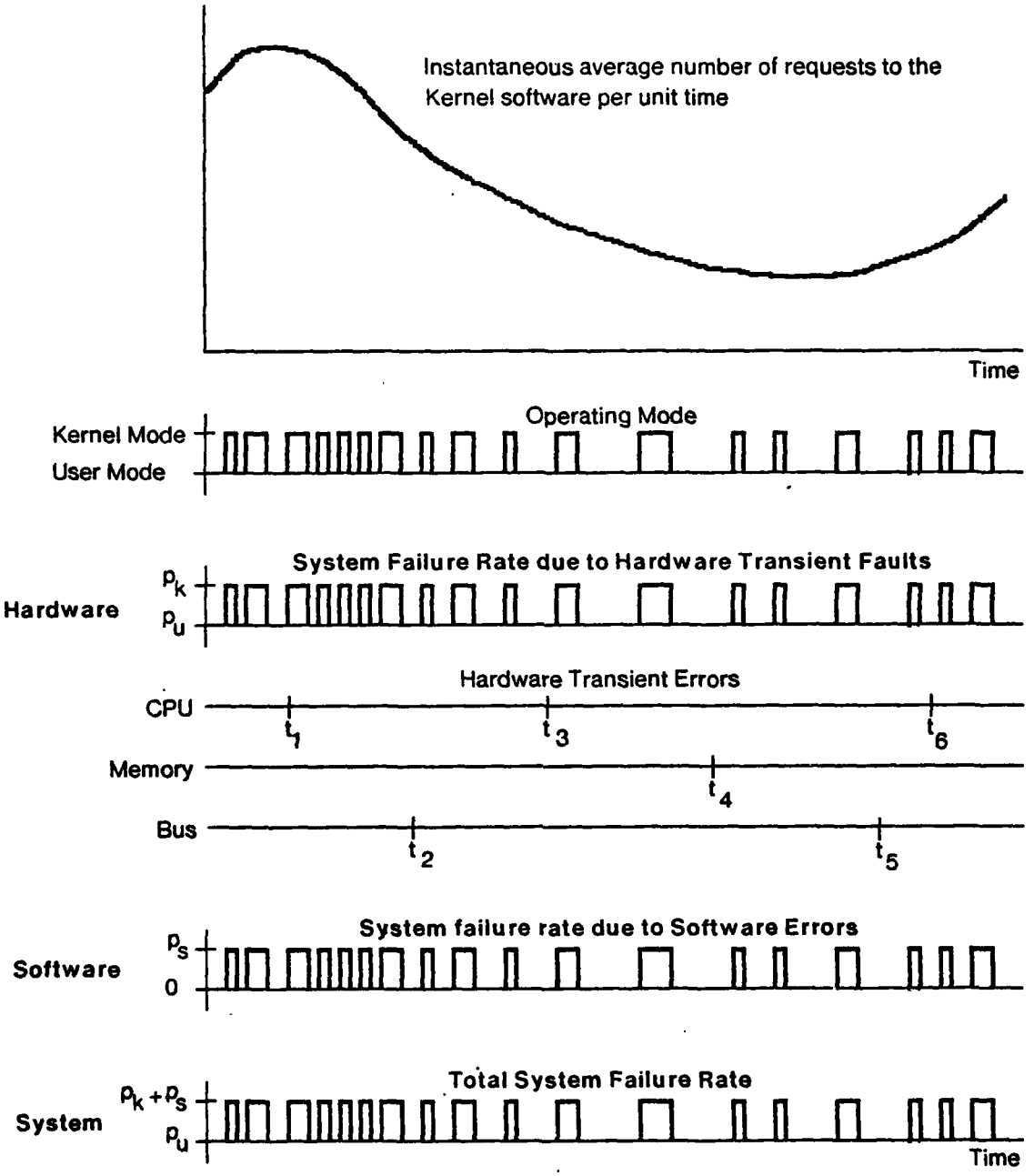


Figure 3-1: Typical sequence of events relevant to the characterization of the reliability of a Time Sharing computer system. System failures due to hardware transients have different probability of leading to a system failure if the system operates in kernel mode than if the system operates in user mode. Kernel software faults can only lead to system failures while parts of the kernel are executed.

Let R_T be the number of requests received by the kernel in a time interval T. A common assumption made in queuing theory is that R_T is a Poisson distributed random variable,

$$P(R_T = n) = \frac{(\nu T)^n}{n!} e^{-\nu T} \tag{3.11}$$

where $P_T(R = n)$ is the probability of receiving n requests in an interval T and ν is the average number of requests received per unit time.

Operational policies and human behavior guarantee that in most Time Sharing systems ν is not going to be a constant but a time varying function reflecting the workload of the system at each time. Thus, more generally, the probability of observing n requests to the kernel in the interval $[t_1,t_2]$ becomes

$$P(R_{[t_1,t_2]} = n) = \frac{\left[\int_{t_1}^{t_2} \nu(\tau)\, d\tau\right]^n}{n!} e^{-\int_{t_1}^{t_2} \nu(\tau)\, d\tau} \tag{3.12}$$

where $\nu(t)$ is the *instantaneous* average number of requests to the kernel per unit time and

$$E[R_{[t_1,t_2]}] = \int_{t_1}^{t_2} \nu_\tau\, d\tau \tag{3.13}$$

is the expected number of requests in the interval $[t_1,t_2]$.

For a doubly stochastic Poisson process, the probability density function (pdf) of the time to the first failure given that the system is started at time $t = t_s$ is given by [Snyder 75]

$$p_{t_f}(\tau|t_s) = E\left[\lambda_\tau e^{-\int_{t_s}^{\tau} \lambda_u\, du}\right] \tag{3.14}$$

where the expectation is taken over the ensemble realizations of λ_τ on $[t_s,\tau]$. As shown in [Saleh 74] the above expectation is equal to

$$p_{t_f}(\tau|t_s) = -\frac{\partial}{\partial \tau} E\left[e^{-\Lambda_{[t_s,\tau]}}\right] \tag{3.15}$$

where

$$\Lambda_{[t_s,\tau]} = \int_{t_s}^{\tau} \lambda_t\, dt \tag{3.16}$$

The statistics of $\Lambda_{[t_s,\tau]}$ are therefore required. Note that the problem of determining the statistics of $\Lambda_{[t_1,t_2]}$ is equivalent to that of finding the distribution of the time that the server of an M/G/1 queue is busy [Kleinrock 75]. The value of $\Lambda_{[t_1,t_2]}$ can be expressed as

$$\Lambda_{[t_1,t_2]} = p_u(t_2-t_1) + (p_k - p_u + p_s) \sum_{i=1}^{R_{[t_1,t_2]}} s_i \tag{3.17}$$

where s_i is the duration needed to serve the i-th request to the kernel. $\Lambda_{[t_1,t_2]}$ is equal to a term that

grows linearly with time plus a random sum of the random variables s_i. Intuitively, for large integration intervals, the expected number terms in the interval will increase such that the distribution of $\Lambda_{[t_1,t_2]}$ should approach a Gaussian distribution. This sum reminds one of the central limit theorem, which roughly states that as the number of terms in a sum of independent and identically distributed random variables approaches infinity, the distribution on the sum approaches the Gaussian distribution. Here though, the number of terms in a time interval is not fixed, but a random variable, and the successive summands may not be independent. However, as will be proved in the following Section, the central limit theorem also holds for $\Lambda_{[t_1,t_2]}$ (under some mild assumptions made precise below). This fact will permit us to use the Gaussian distribution to compute expectations of the type shown in (3.15).

A stronger limit theorem can also be proved for the distribution of $\Lambda_{[t_1,t_2]}$. The integral of the failure rate process converges, in fact, to a Wiener process. This result, proved in the Section 3.2.3, will allow us to explain why the apparent hazard function of the failure process is a decreasing function of time and will permit us to compute its limiting value. Curiously, the rate at which $\Lambda_{[t_1,t_2]}$ converges to a Wiener process will be shown to be one of the parameters characterizing the reliability of such complex systems as Time Sharing computers.

REMARK: If ν is a constant independent of time, all the parameters characterizing the underlying intensity process are constant and λ_t will be stationary. Under this assumption, the failure process becomes a renewal process. Indeed, no repair takes place after either transients or software faults. Therefore, after each failure the system is restarted and starts operating as new.

3.2.2. The central limit theorem for a random sum of dependent variables

$\Lambda_{[t_1,t_2]}$ can be rewritten as

$$\Lambda_{[t_1,t_2]} = \alpha(t_2-t_1) + \beta\sum_{i=1}^{R_{[t_1,t_2]}} s_i \qquad (3.18)$$

where $\alpha = p_u$ and $\beta = (p_k - p_u + p_s)$. $R_{[t_1,t_2]}$ is the number of requests to the kernel in the interval $[t_1,t_2]$ and it is assumed to be a Poisson distributed random variable with pdf given in (3.12). s_i is the time required to satisfy the i-th request. The s_i will be assumed to be identically distributed. It cannot be assumed, though, that they are mutually independent since requests to the kernel close in time are likely to be related one way or another (e.g., only a process that has been recently activated can be deactivated). However, it is reasonable to assume that requests to the kernel separated by a long time are independent. Thus, the sequence $\{s_i\}$ will be assumed to be stationary and α-mixing, two concepts that are defined as follows :

Definition 20: Given a sequence $\{s_i\}$ of random variables, the sequence is said to be α-mixing if there exists a sequence $\{\alpha_n\}$ such that for each k,

$$\left|P(A\cap B) - P(A)P(B)\right| \leq \alpha_n \qquad \text{and } \alpha_n \to 0 \text{ as } n \to \infty \tag{3.19}$$

$$A \epsilon \sigma(s_1, \ldots\ldots, s_k)$$

$$B \epsilon \sigma(s_{k+n}, s_{k+n+1}, \ldots\ldots)$$

Definition 21: A sequence $\{s_i\}$ is said to be *stationary* if the distribution of the random vector $(s_i, s_{i+1}, \ldots\ldots, s_{i+k})$ does not depend on i.

It is therefore assumed that s_i and s_{i+k} are approximately independent for large k and the statistics of $\{s_i\}$ are independent of the time origin. Define now

$$x_i = s_i - E[s_i] \tag{3.20}$$

and let

$$S_k = \sum_{i=1}^{k} x_i \tag{3.21}$$

such that $E[S_k] = E[x_i] = 0$. Without loss of generality assume $t_1 = 0$, $t_2 = t$, $\alpha = \beta = 1$. The integral of the failure rate process can now be expressed as

$$\Lambda_t = t + R_t E[s_i] + S_{R_t} \tag{3.22}$$

Let the following conditions be defined :

Condition 1: *Convergence of* $\{S_k\}$. If

$$P\left\{ \frac{S_n}{n^{1/2}\sigma} \leq \alpha \right\} \Rightarrow \Phi(\alpha) \tag{3.23}$$

The sequence $\{S_k\}$ is then said to satisfy the central limit theorem with norming factors $n^{1/2}\sigma$.

Condition 2: *Uniform continuity* of $\{S_k\}$. Given any small positive ε and η, there is a large n_0 and a small positive δ such that if $n \geq n_0$ then

$$P\left\{ \max_{|k-n| \leq \delta} |S_n - S_k| < \varepsilon n^{1/2}\sigma \right\} > 1-\eta \tag{3.24}$$

Condition 3: *Convergence* of R_t. Let R_t be a sequence of integer valued random variables such that

$$\operatorname{p\,lim}_{n\to\infty} \frac{R_t}{t} = \nu \tag{3.25}$$

Anscombe has proved the following result [Anscombe 52]

Theorem 22: *Suppose that* $\{S_k\}$ satisfies the central limit theorem with norming factors $n^{1/2}\sigma$ (Condition 1), that the convergence is uniform in probability (Condition 2) and that R_t is an integer valued random variable satisfying *Condition* 3. *Then*

$$P\left\{ \frac{S_{R_t}}{(t\nu)^{1/2}\sigma} \le \alpha \right\} \Rightarrow \Phi(\alpha) \tag{3.26}$$

That is, the central limit theorem also holds for a sequence in which the number of summands is a random variable provided that Conditions 1 through 3 are satisfied.

A proof that the $\{S_k\}$ satisfies the central limit theorem when the x_i form a stationary, α-mixing sequence can be found in [Billingsley 79], a fact that is stated precisely in the following theorem.

Theorem 23: Suppose that $\{x_i\}$ is a stationary, α-mixing sequence with $\alpha_n = O(n^{-5})$, and that $E[x_i] = 0$ and $E[x_i^{12}] < \infty$. If

$$S_k = \sum_{i=1}^{k} x_i \tag{3.27}$$

then $\{S_k\}$ satisfies the central limit theorem with norming factors $n^{1/2}\sigma$ where

$$\frac{\mathrm{Var}[S_n]}{n} \to \sigma^2 = E[x_1^2] + 2\sum_{k=1}^{\infty} E[x_1 x_{1+k}] \tag{3.28}$$

and the series converges absolutely. If $\sigma > 0$, then

$$P\left\{ \frac{S_n}{n^{1/2}\sigma} \le \alpha \right\} \Rightarrow \Phi(\alpha) \tag{3.29}$$

Thus, a stationary, α-mixing sequence satisfies Condition 1. A Poisson distributed random variable obviously satisfies Condition 3 (provided that $\nu(t)$ is bounded). To use Anscombe's theorem it is necessary to verify that a stationary, α-*mixing sequence* is uniformly continuous in probability (Condition 2).

Lemma 24: Suppose $\{x_i\}$ to be a stationary, α-mixing sequence with $\alpha_n = O(n^{-5})$ and let $\{S_k\}$ be the sequence defined in (3.27). Then, given any small ε and η there exits a small positive δ and a large n_0 such that *if* $n > n_0$

$$P\left\{ \max_{|k-n| \le \delta n} |S_n - S_k| < \varepsilon n^{1/2}\sigma \right\} > 1-\eta \tag{3.30}$$

PROOF: It must be shown that given $\varepsilon, \eta > 0$ there is a $\delta > 0$ such that for $n > n_0$

$$P\{ |S_n - S_k| \geq \varepsilon n^{1/2}\sigma \} < \eta \quad \text{for any k such that } |k-n| \leq \delta n \tag{3.31}$$

In particular, it will be shown that

$$P\{ |S_n - S_k| \geq \varepsilon n^{1/2}\sigma \} \quad \text{for any k such that } n \leq k \leq (1+\delta)n \tag{3.32}$$

can be made arbitrarily small. A similar argument would apply to values of k where $(1-\delta)n \leq k \leq n$. By Tchebyschev's inequality,

$$P\left\{ \left| \sum_{i=n+1}^{k} x_i \right| \geq \varepsilon n^{1/2}\sigma \right\} \leq \frac{\mathrm{Var}\left[\sum_{i=n+1}^{k} x_i\right]}{\varepsilon^2 n\sigma^2} \tag{3.33}$$

$$P\left\{ \left| \sum_{i=n+1}^{(1+\delta)n} x_i \right| \geq \varepsilon n^{1/2}\sigma \right\} \leq \frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i\right]}{\varepsilon^2 n\sigma^2} \tag{3.34}$$

where, by stationarity

$$\mathrm{Var}\left[\sum_{i=n+1}^{k} x_i\right] = (k-n)E[x_1^2] + 2\sum_{i=1}^{k-n-1}(k-n-i)\,E[x_1 x_{1+i}] \tag{3.35}$$

$$\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i\right] = \delta n E[x_1^2] + 2\sum_{i=1}^{\delta n-1}(\delta n-i)\,E[x_1 x_{1+i}] \tag{3.36}$$

Thus,

$$\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i\right] \cdot \mathrm{Var}\left[\sum_{i=n+1}^{k} x_i\right]$$
$$= (\delta n + n - k)E[x_1^2] + 2(\delta n + n - k)\sum_{i=1}^{k-n-1}E[x_1 x_{1+i}] + 2\sum_{k-n}^{\delta n-1}(\delta n - i)E[x_1 x_{1+i}] \tag{3.37}$$

Since $n \leq k \leq (1+\delta n)$, let $k'n = k$, $1 \leq k' \leq 1+\delta$. From the properties of an α-mixing sequence,

$$\lim_{n\to\infty} \sum_{(k'-1)n}^{\delta n-1} (\delta - i/n)E[x_1 x_{1+i}] = 0 \tag{3.38}$$

and

$$\lim_{n\to\infty} \frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i\right] \cdot \mathrm{Var}\left[\sum_{i=n+1}^{k} x_i\right]}{n} =$$
$$= (\delta + 1 - k')E[x_1^2] + 2(\delta + 1 - k')\sum_{i=1}^{\infty} E[x_1 x_{1+i}] \tag{3.39}$$

The series converges absolutely for the same reason that σ^2 converges absolutely in (3.28) (see [Billingsley 79] for details). Thus, if $\sigma > 0$, the above limit is also positive. Therefore, there exists an n_0 such that

$$\frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i\right] \cdot \mathrm{Var}\left[\sum_{i=n+1}^{K} x_i\right]}{n} > 0 \qquad n > n_0 \tag{3.40}$$

Hence, for $n > n_0$,

$$P\left\{ \left| \sum_{i=n+1}^{k} x_i \right| \geq \varepsilon n^{1/2}\sigma \right\} \leq \frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i \right]}{\varepsilon^2 n\sigma^2} \tag{3.41}$$

But

$$\frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i \right]}{n} = \delta E[x_i^2] + 2\sum_{i=1}^{\delta n-1} (\delta - \frac{i}{n}) E[x_1 x_{1+i}] \tag{3.42}$$

which, given $n>n_0$, can be made arbitrarily small by a proper choice of δ. In particular, chose δ such that

$$\frac{\mathrm{Var}\left[\sum_{i=n+1}^{(1+\delta)n} x_i \right]}{n} < \eta\varepsilon^2\sigma^2 \tag{3.43}$$

and (3.30) follows. ∎

It is now possible to prove the following theorem :

Theorem 25: Let $\{x_i\}$ be a stationary and α-mixing sequence of random variables with $\alpha_n = O(n^{-5})$, and let N_t be a sequence of Poisson distributed random variables satisfying Condition 3. Then,

$$P\left\{ \frac{S_{N_t}}{(t\nu)^{1/2}\sigma} \leq \alpha \right\} \Rightarrow \Phi(\alpha) \tag{3.44}$$

where σ has been given in (3.28).

PROOF : Condition 1 holds by Theorem 23. Condition 2 holds by Lemma 24. Condition 3 holds for a Poisson distributed random variable if $\nu(t)$ is bounded. Therefore, by Theorem 22 the limit of the random sum converges to the Gaussian distribution. ∎

Corollary 26: Let $N_{[t_1,t_2]}$ be a doubly stochastic Poisson process with intensity process λ_t as defined in (3.10). If

$$\operatorname{p\,lim}_{t\to\infty} \frac{1}{t} \int_0^t \nu(\tau)\, d\tau = \nu \tag{3.45}$$

then, the Probability Distribution Function (PDF) of the time to the first failure given that the system is under observation since time t_s is given by

$$P(t_f<\tau|t_s) = e^{-E\left[\Lambda_{[t_s,\tau]}\right] + \frac{\mathrm{Var}\left[\Lambda_{[t_s,\tau]}\right]}{2}} \tag{3.46}$$

where

$$E\left[\Lambda_{[t_s,\tau]}\right] = \alpha(\tau - t_s) + \beta\, E[R_{[t_s,\tau]}]\, E[s_i] \tag{3.47}$$

$$\mathrm{Var}\left[\Lambda_{[t_s,\tau]}\right] = \beta^2 \left(E[R_{[t_s,\tau]}]\sigma^2 + E[s_i]^2\, \mathrm{Var}[R_{[t_s,\tau]}] \right) \tag{3.48}$$

PROOF : Remember that

$$\Lambda_{[t_1,t_2]} = \alpha(t_2-t_1) + \beta \left(R_{[t_1,t_2]}E[s_i] + S_{R_{[t_1,t_2]}} \right) \tag{3.49}$$

where $x_i = s_i - E[s_i]$ and $S_k = \sum_{i=1}^{k} x_i$. For large $[t_1,t_2]$, by Theorem 25 $S_{R_{[t_1,t_2]}}$ converges in distribution to a Gaussian random variable with zero mean and variance

$$Var[S_{R_{[t_1,t_2]}}] = E[R_{[t_1,t_2]}]\sigma^2 \tag{3.50}$$

where σ is given in (3.28). $R_{[t_1,t_2]}$ is a Poisson distributed random variable. Hence, for large $[t_1,t_2]$ it is also asymptotically normal with mean and variance

$$E[R_{[t_1,t_2]}] = Var[R_{[t_1,t_2]}] = \int_{t_1}^{t_2} \nu(\tau)\, d\tau \tag{3.51}$$

Furthermore,

$$E\left[R_{[t_1,t_2]}S_{R_{[t_1,t_2]}}\right] = \sum_{r=1}^{\infty} rP(R_{[t_1,t_2]} = r)\, E[S_r] = 0 \tag{3.52}$$

Therefore,

$$E\left\{e^{-\Lambda_{[t_1,t_2]}}\right\} = e^{-\alpha(t_2-t_1)}\, E\left\{e^{-\beta E[s_i]R_{[t_1,t_2]}}\right\}\, E\left\{e^{-\beta S_{R_{[t_1,t_2]}}}\right\} \tag{3.53}$$

Note that if z is a Gaussian distributed random variable with mean m_z and variance σ_z^2

$$E\{e^{-z}\} = \frac{1}{(2\pi)^{1/2}\sigma_z} \int_{-\infty}^{\infty} e^{-z} e^{-\frac{(z-m_z)^2}{2\sigma_z^2}}\, dz \tag{3.54}$$

$$= e^{-m_z + \frac{\sigma_z^2}{2}} \tag{3.55}$$

Hence,

$$E\left\{e^{-\beta E[s_i]R_{[t_1,t_2]}}\right\} = \exp\left\{-\beta E[R_{[t_1,t_2]}]E[s_i] + \frac{\beta^2 E[s_i]^2 Var[R_{[t_1,t_2]}]}{2}\right\} \tag{3.56}$$

and

$$E\left\{e^{-\beta S_{R_{[t_1,t_2]}}}\right\} = \exp\left\{\frac{\beta^2 E[R_{[t_1,t_2]}]\sigma^2}{2}\right\} \tag{3.57}$$

and (3.46) follows. ∎

The distribution of $\{S_k\}$ not only converges to the Gaussian distribution for a large (expected) number of summands, but $\{S_k\}$ also satisfies the so called *invariance principle*, a concept for which some more elaborate mathematical tools are required and which is described in the following Section.

3.2.3. Convergence to Wiener measure

Let C be the space of continuous functions on [0,1] and let $\mathcal{C}$ be its σ-field of Borel sets. For each $\omega\in\Omega$ let $p(u) = p(u,\omega)$ be the function defined on $[0,\infty)$ defined by

$$p(u) = S_{\lfloor u\rfloor} + (u-\lfloor u\rfloor)x_{\lfloor u\rfloor+1} \qquad (3.58)$$

For $n = 1,2,\ldots$ define $p_n(u) = p_n(u,\omega)$ for $0\leq t\leq 1$ by

$$p_n(t) = \frac{p(tn)}{n^{1/2}\sigma} \qquad (3.59)$$

Thus, $p_n(\cdot)$ is that element of C which is linear on each interval $\left[(k-1)/n\ ,\ k/n\right]$ and satisfies

$$p_n(k/n) = \frac{S_k}{n^{1/2}\sigma} \qquad k\leq n \qquad (3.60)$$

Definition 27: If $P_n(A) = P\{p_n\in A\}$ for $A\in\mathcal{C}$ then we say that $\{x_i\}$ satisfies the *invariance principle* with norming factors $n^{1/2}\sigma$ if $P_n(A)\Rightarrow\mathcal{W}(A)$, where $\mathcal{W}(\cdot)$ denotes Wiener measure.

Now, for integers c,ν,n define $n_j = jn/c$ $j = 0,1,\ldots,c$ and $n_{j,u} = n(\nu(j-1)+u)/c\nu$, $j = 1,\ldots,c$ $n = 0,1,\ldots,\nu$.

Definition 28: For any real numbers α_j, β_j let $E_{n,r}$ be the set in Ω where the relations

$$\alpha_j \leq \frac{S_i}{n^{1/2}\sigma} \leq \beta_j \qquad \text{if } n_{j-1}<i\leq n_j \qquad (3.61)$$

are satisfied for $i<r$ but not for $i = r$.

Define the following two conditions :

Condition 4: For any integer c

$$\lim_{n\to\infty} P\left\{ S_{in/c}-S_{(i-1)n/c} \leq \alpha_i(nc)^{1/2}\sigma \right\} = \prod_{i=1}^{c}\Phi(\alpha_i) \qquad i = 1,\ldots,c \qquad (3.62)$$

Condition 5: *For any integer c, any set $(\alpha_1,\ldots,\alpha_c,\beta_1,\ldots,\beta_c)$ and each $\varepsilon>0$*

$$\lim_{\nu\to\infty}\ \limsup_{n\to\infty} \sum_{r=1}^{n} P\left(E_{n,r}\cap\left\{\left|S_{r'}-S_r\right|\geq\varepsilon n^{1/2}\sigma\right\}\right) = 0 \qquad (3.63)$$

where $r' = n_{j,u+1}$ is that integer such that $n_{j,u}<r\leq n_{j,u+1}$.

The conditions under which the invariance principle holds for sequences of dependent random variables are stated now. A proof of the following theorem can be found in [Billingsley 56]

Theorem 29: The invariance principle holds for the sequence $\{S_i\}$ if Conditions 4 and 5 are satisfied.

It will now be proved that the invariance principle holds for a stationary, α-mixing sequence of random variables.

Theorem 30: Let $\{x_i\}$ be a sequence of stationary, α-mixing random variables with $\alpha_n = O(n^{-5})$ and $E[|x_i|^4] < \infty$. The invariance principle holds for $\{S_k\}$ with *norming* factors $(nc)^{1/2}\sigma$, where σ has been defined in (3.28)

PROOF : It must be proved that a stationary and α-mixing sequence satisfies Conditions 4 and 5. Condition 4 will be proved first. By stationarity,

$$P\left\{ S_{in/c} - S_{(i-1)n/c} \le \alpha_i n^{1/2}\sigma \right\} = P\left\{ \sum_{i=1}^{n/c} x_i \le \alpha_i n^{1/2} c\sigma \right\} \tag{3.64}$$

$$= P\left\{ \sum_{i=1}^{m} x_i \le \alpha_i (mc)^{1/2}\sigma \right\} \tag{3.65}$$

Therefore, by Theorem 23

$$\lim_{n\to\infty} P\left\{ S_{in/c} - S_{(i-1)n/c} \le \alpha_i (nc)^{1/2}\sigma \right\} = \Phi(\alpha_i) \tag{3.66}$$

Define now

$$\mathbb{C}_n^{i,j} = E\left[\left(S_{in/c} - S_{(i-1)n/c}\right)\left(S_{jn/c} - S_{(j-1)n/c}\right) \right] \tag{3.67}$$

Also by stationarity, assuming $j > i$,

$$\mathbb{C}_n^{i,j} = E\left[\left(\sum_{i=1}^{n/c} x_i\right)\left(\sum_{((j-i)n/c)+1}^{(j-i+1)n/c} x_i\right) \right] \tag{3.68}$$

By the definition of an α-mixing sequence,

$$\mathbb{C}_n^{i,j} \le E\left[\sum_{i=1}^{n/c} x_i\right] E\left[\sum_{((j-i)n/c)+1}^{(j-i+1)n/c} x_i\right] + \alpha_{(j-i)n/c} \tag{3.69}$$

and

$$\mathbb{C}_n^{i,j} \ge E\left[\sum_{i=1}^{n/c} x_i\right] E\left[\sum_{((j-i)n/c)+1}^{(j-i+1)n/c} x_i\right] - \alpha_{(j-i)n/c} \tag{3.70}$$

Since $E[x_i] = 0$ and

$$\lim_{n\to\infty} \alpha_{(j-i)n/c} = 0 \tag{3.71}$$

it follows that

$$\lim_{n\to\infty} \mathbb{C}_n^{i,j} = 0 \qquad i \neq j \tag{3.72}$$

and

$$\lim_{n\to\infty} \frac{\mathbb{C}_n^{i,j}}{n} = E\left[\left(\sum_{j=1}^{n/c} x_j \right)^2 \right] \tag{3.73}$$

$$= c\sigma^2 \tag{3.74}$$

Thus, as $n \to \infty$ the distribution of the random vector

$$\frac{1}{(nc)^{1/2}\sigma}\left(S_{n_1},S_{n_2}-S_{n_1},\ldots\ldots,S_{n_c}-S_{n_{c-1}}\right) \tag{3.75}$$

approaches a multidimensional Gaussian distribution having as covariance matrix the identity matrix and Condition 4 is satisfied. As for Condition 5, note that

$$P\left(E_{n,r}\cap\left\{\left|S_{r'}-S_r\right|\geq\varepsilon n^{1/2}\sigma\right\}\right)\leq P\left(E_{n,r}\cap\left\{\left|S_{r'}-S_{r+m}\right|\geq\varepsilon n^{1/2}\sigma/2\right\}\right)$$
$$+P\left(E_{n,r}\cap\left\{\left|S_{r+m}-S_r\right|\geq\varepsilon n^{1/2}\sigma/2\right\}\right) \tag{3.76}$$

As for the second term in the right hand of (3.76),

$$P\left\{E_{n,r}\cap\left|S_{r+m}-S_r\right|\geq\varepsilon n^{1/2}\sigma/2\right\}=P\left\{\left|S_{r+m}-S_r\right|\geq\varepsilon n^{1/2}\sigma/2\right\} \tag{3.77}$$
$$=P\left\{\left|\sum_{\nu=r}^{r+m}x_\nu\right|\geq\varepsilon n^{1/2}\sigma/2\right\} \tag{3.78}$$
$$\leq\sum_{\nu=r}^{r+m}P\left\{\left|x_\nu\right|\geq\varepsilon n^{1/2}\sigma/2m\right\} \tag{3.79}$$

Hence,

$$\sum_{r=1}^{n}P\left\{\left|S_{r+m}-S_r\right|\geq\varepsilon n^{1/2}\sigma/2\right\}\leq m\sum_{\nu=1}^{n}P\left\{\left|x_\nu\right|\geq\varepsilon n^{1/2}\sigma/2m\right\} \tag{3.80}$$
$$\leq m\left(\frac{2m}{\varepsilon\sigma}\right)^{2+\delta}\frac{1}{n^{1+\delta/2}}\sum_{\nu=1}^{n}E\left\{\left|x_\nu\right|^{2+\delta}\right\} \tag{3.81}$$

for any $\delta>0$ by Tchebyshev's inequality. Chose now $\delta=2$ and $m=O(n^{1/5})$ and

$$\lim_{n\to\infty}\sum_{r=1}^{n}P\left\{\left|S_{r+m}-S_r\right|\geq\varepsilon n^{1/2}\sigma/2\right\}=0 \tag{3.82}$$

And now for the first term in (3.76). By the properties of an α-mixing sequence and since $E_{n,r}$ is defined in $\sigma(x_1,\ldots,x_r)$

$$\sum_{r=1}^{n}P\left(E_{n,r}\cap\left\{\left|S_{r'}-S_{r+m}\right|\geq\varepsilon n^{1/2}\sigma/2\right\}\right)$$
$$\leq\sum_{r=1}^{n}P(E_{n,r})\,P\left\{\left|S_{r'}-S_{r+m}\right|\geq\varepsilon n^{1/2}\sigma/2\right\}+\alpha_m \tag{3.83}$$
$$\leq\left(\max_{r}P\left\{\left|S_{r'}-S_{r+m}\right|\geq\varepsilon n^{1/2}\sigma/2\right\}\right)+\alpha_m \tag{3.84}$$
$$\leq\left(\max_{r}\frac{4\,\mathrm{Var}\left[\left|S_{r'}-S_{r+m}\right|\right]}{\varepsilon^2n\sigma^2}\right)+\alpha_m \tag{3.85}$$
$$\leq\frac{4\xi_1}{\varepsilon^2c\nu}+\frac{4m\xi_2}{\varepsilon^2nc\nu} \tag{3.86}$$

where ξ_1 and ξ_2 are bounded. The last inequality has been obtained taking into account that $r'-r$ will be at most $n/c\nu$ and rewriting the variance of $|S_{r'}-S_{r+m}|$ as a function of σ^2. Therefore

$$\limsup_{n\to\infty} \sum_{r=1}^{n} P\left(E_{n,r} \cap \left\{ |S_r - S_{r+m}| \geq \epsilon n^{1/2}\sigma/2 \right\} \right) \leq \frac{4\xi_1}{\epsilon^2 c\nu} \tag{3.87}$$

As $n\to\infty$, the second term in the right hand side of (3.76) goes to 0 by (3.82) .As $\nu\to\infty$ the first term also goes to 0 by (3.87), and (3.63) follows. ∎

Since the sequence $\{x_i\}$ satisfies the invariance principle it is now possible to use the following theorem also due to [Billingsley 63]. Let R_t be a sequence of integer valued random variables. For a realization of the sequence, $R_t(\omega)$ let $\xi_1(\omega)$, $\xi_2(\omega)$,.... be the successive discontinuities of $R_t(\omega)$ as a function of t, so that $R_t = i$ if $\xi_i \leq t \leq \xi_{i+1}$. Define now

$$R'_t = i + \frac{t-\xi_i}{\xi_{i+1}-\xi_i} \qquad \text{if } \xi_i \leq t < \xi_{i+1} \tag{3.88}$$

Thus, R'_t is that function of t which is linear on each interval $[\xi_i, \xi_{i+1}]$ and agrees with R_t at its jumps. Define now $q(t) = p(R'_t)$, where $p(\cdot)$ has been defined in (3.58), and $q_u(t) = q(ut)/(\nu u)^{1/2}\sigma$. Define a measure on $\mathcal{C}$ by $Q_u(A) = P\{q_u \in A\}$.

Theorem 31: If $\{S_i\}$ satisfies the invariance principle with norming factors $n^{1/2}\sigma$ and

$$p\lim_{t\to\infty} \left\{ \sup_{\tau\leq t} \left| \frac{R_\tau - \nu\tau}{t} \right| \right\} = 0 \tag{3.89}$$

Then $\{S_{R_t}\}$ satisfies the invariance principle, that is, $Q_u(A) \Rightarrow \mathcal{W}(A)$.

It is now possible to prove the convergence of $\Lambda_{[t_1,t_2]}$ to a Wiener process.

Corollary 32: Let λ_t be the failure rate process defined in (3.10). As the integration interval approaches infinity, the integrated rate Λ_t converges in distribution to a Wiener process W_t with

$$E[W_t] = (\alpha + \beta E[s_i]\nu)t \tag{3.90}$$

$$\mathrm{Var}[W_t] = \beta^2\nu(\sigma^2 + E[s_i]^2)t \tag{3.91}$$

PROOF : The proof is identical to that of Corollary 26. Just note that R_t converges also to a Wiener process independent of S_{R_t}. Further, note that (3.89) is satisfied since

$$P\left\{ \sup_{\tau\leq t} \left| \frac{R_\tau - \nu\tau}{t} \right| > \epsilon \right\} \leq \frac{\mathrm{Var}[R_t]}{\epsilon^2 t^2} \tag{3.92}$$

$$\leq \frac{\nu t}{\epsilon^2 t^2} \tag{3.93}$$

which goes to zero as $t\to\infty$. ∎

By the definition of Wiener measure,

$$P\left\{ \frac{S_{R_{ut}}}{(u)^{1/2}\sigma} \leq \alpha \right\} \Rightarrow \mathcal{W}(t,\alpha) = \frac{1}{(2\pi t)^{1/2}} \int_{-\infty}^{\alpha} e^{-x^2/2t} dx$$

Hence, the invariance principle implies the central limit theorem. However, the invariance principle is a much stronger limit as it also implies that Λ_t has independent increments. That is, as u approaches infinity the random variables

$$S_{R_{u[t_{k-1},t_k]}} \quad \text{and} \quad S_{R_{u[t_k,t_{k+1}]}}$$

are independent, normally distributed random variables. This result could never be obtained from the central limit theorem.

3.3. The observable process

In the previous section the PDF of the time to failure of a computing system has been characterized by some convergence limits. The expressions obtained depend on some statistical properties of the time that the kernel operates in kernel mode. In particular, they depend on the variance

$$\sigma^2 = E[x_1^2] + 2\sum_{k=1}^{\infty} E[x_1 x_{1+k}] \tag{3.94}$$

where the x_i are the service times of successive requests to the kernel. Unfortunately, the measurement of $E[x_1 x_{1+k}]$ is not likely to be possible on real systems. The kernel is executed at least once per line clock tick, 60 clock ticks per second. To estimate the above statistic, either a complex hardware monitor is required or the entire kernel software has to be modified such that at the start and end of each service a time stamp is recorded somewhere. Both approaches are cumbersome and impractical for operational, commercially available computing systems. Since one of the premises of the present work is that any mathematical characterization must be verifiable from easily measurable variables in operating computers, an alternate way is required.

3.3.1. The observable intensity process

Let the process $\bar{\lambda}_t$ be defined as follows :

$$\bar{\lambda}_t = \frac{1}{W} \int_{t-W/2}^{t+W/2} \lambda_\tau \, d\tau \tag{3.95}$$

that is, $\bar{\lambda}_t$ is the result of averaging λ_t over an interval of duration W. The question now is

$$\int_{t_1}^{t_2} \bar{\lambda}_\tau \, d\tau \stackrel{?}{\simeq} \int_{t_1}^{t_2} \lambda_\tau \, d\tau$$

If the integral of λ_t can be approximated by the integral of $\bar{\lambda}_t$ the situation is much better. Most operating systems automatically measure the cumulative time in kernel mode, such that the values of $\bar{\lambda}_t$ can be easily sampled. Fortunately, the answer is affirmative. The exact value of the integral of $\bar{\lambda}_t$ is

$$\int_{t_1}^{t_2} \bar{\lambda}_\tau \, d\tau = \frac{1}{W} \int_{t_1}^{t_2} \int_{t-W/2}^{t+W/2} \lambda_\tau \, d\tau \, dt \tag{3.96}$$

$$= \int_{t_1-W/2}^{t_2+W/2} W^{w}_{[t_1,t_2]}(\tau) \, \lambda_t \, d\tau \tag{3.97}$$

where $W^{w}_{[t_1,t_2]}$ is the window function shown in Figure 3-2.

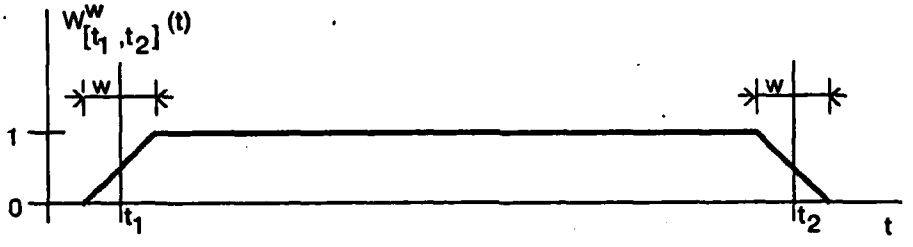


Figure 3-2: Window function used to obtain the integral of $\bar{\lambda}_t$

That is,

$$\int_{t_1}^{t_2} \bar{\lambda}_\tau \, d\tau = \int_{t_1}^{t_2} \lambda_\tau \, d\tau + e_w \tag{3.98}$$

The absolute value of the error term depends only on w. As the integration interval $[t_1,t_2]$ increases, the error term remains constant.

Given a realization of λ_t, $\tilde{\Lambda}_t$ is defined such that

$$\tilde{\Lambda}_{[t_1,t_2]} = \int_{t_1}^{t_2} \bar{\lambda}_\tau \, d_t \tag{3.99}$$

can be used as an approximation of $\Lambda_{[t_1,t_2]}$. Thus, the pdf of the time to failure can be approximated by

$$p(\tau|t_0) = -\frac{\partial}{\partial \tau} E\left\{ e^{-\tilde{\Lambda}_{[t_1,t_2]}} \right\} \tag{3.100}$$

Now let the value of the averaging window, w, be sufficiently large that the central limit theorem holds for $\tilde{\lambda}_t$. For fixed t, $\tilde{\lambda}_t$ can be approximated by a Gaussian random variable. This assumption is consistent. Let w = 10 sec. $\tilde{\lambda}_t$ is then equal to the sum of $\sim 10^3$ random variables. But typical MTTF values are on the order of, at least, hours. Hence, the evaluation of $\tilde{\Lambda}_t$ will be based on an integral over, say, 10 hours. The error term is equal to an integral over a period of 10 sec., and therefore can be neglected.

$\tilde{\lambda}_t$ then becomes a Gaussian stochastic process, and $\tilde{\Lambda}_{[t_1,t_2]}$, being the integral of a Gaussian process over a finite interval, will obviously be a Gaussian random variable. If $\tilde{\lambda}_t$ is a Gaussian stochastic process with mean $E[\tilde{\lambda}]$ and autocorrelation function $R_{\tilde{\lambda}\tilde{\lambda}}(s,t)$, $\tilde{\Lambda}_{[t_1,t_2]}$ is a Gaussian random variable with mean

$$E[\tilde{\Lambda}_{[t_1,t_2]}] = \int_{t_1}^{t_2} E[\tilde{\lambda}_t]\, dt \tag{3.101}$$

and variance

$$Var[\tilde{\Lambda}_{[t_1,t_2]}] = 2\int_{t_1}^{t_2}\int_{t_1}^{t_2} \left[R_{\tilde{\lambda}\tilde{\lambda}}(s,t) - E[\tilde{\lambda}_s]E[\tilde{\lambda}_t]\right] ds\, dt \tag{3.102}$$

(see [Papoulis 65], pp. 323:325). Hence,

$$P(t_f<\tau|t_s) = e^{-E\left[\tilde{\Lambda}_{[t_s,\tau]}\right] + \frac{Var\left[\tilde{\Lambda}_{[t_s,\tau]}\right]}{2}} \tag{3.103}$$

The difference between (3.103) and (3.46) is that the values of (3.101) and (3.102) are much easier to estimate from an operational system than the values of (3.47) and (3.48). To estimate (3.101) and (3.102) all it is needed is a sequence of sample values of the fraction of time in kernel mode. And this *is* an easily observable sequence.

3.4. The equivalent failure process

Expression (3.103) gives the PDF of the time to the first failure given that the system is observed starting at time t_0. Given $E[\tilde{\lambda}_t]$ and $R_{\tilde{\lambda}\tilde{\lambda}}(s,t)$ all the functions on which $P(t_f \leq \tau|t_s)$ depends are known and deterministic. Expression (3.103) can therefore be viewed as the PDF of a nonhomogeneous Poisson process.

REMARK : The fact the the PDF of the failure process introduced in Sections 3.2 and 3.3 is equivalent to the PDF of a nonhomogeneous Poisson process with PDF given in (3.103) does not

mean that the two processes are indistinguishable. It only means that the statistics of the time to the first failure are indistinguishable. However, if λ_t is stationary such that the failure process is a renewal process, then the process with stochastic intensity and the nonhomogeneous Poisson process are truly equivalent.

A nonhomogeneous Poisson process is a much simpler conceptual framework to work with than the situation described in the previous sections of this chapter. A nonhomogeneous Poisson process is completely characterized by its hazard function, a deterministic, time varying function. Thus, if reliability characterization can be made based only on the distribution of the time to failure, the hazard function of the equivalent failure process is all that is needed.

3.4.1. The hazard function

A nonhomogeneous Poisson process is completely specified from its hazard function. From (3.5) note that

$$h(t) = \frac{p(t)}{1-P(t_f \leq t)} \tag{3.104}$$

Thus, from (3.103),

$$h(t) = \frac{\partial E[\tilde{\Lambda}_t]}{\partial t} - \frac{1}{2}\frac{\partial Var[\tilde{\Lambda}_t]}{\partial t} \tag{3.105}$$

From (3.90), (3.91), and (3.105) it can be seen that for large integration intervals the hazard function becomes a constant independent of time. If the system has been started at $t = 0$, the *quantity* $h(t)\Delta t$ is the probability that a failure will occur in the interval $[t, t + \Delta t)$. Therefore, as the integrated failure rate converges to a Wiener process, the hazard function of the equivalent process converges to a constant independent of time. In that case, the equivalent failure process degenerates to a homogeneous Poisson process. For a homogeneous Poisson process, the number of points in disjoint time intervals are statistically independent. This in turn implies that the random variables

$$\Lambda_1 = \int_{t_0}^{t_1} \lambda_\tau \, d\tau \qquad \text{and} \qquad \Lambda_2 = \int_{t_1}^{t_2} \lambda_\tau \, d\tau$$

are statistically independent. Therefore, the convergence to a constant hazard function could not have been guessed from the central limit theorem alone since in general, Λ_1 and Λ_2 will not be independent being the sum of an α-mixing sequence.

3.5. Summary

This chapter started with the assumption that the failure rate of a Time Sharing computer is continuously switching between two states. While in each state, the system has a given sensitivity to the presence of hardware transient faults and software faults. The PDF of the time to failure depends on the integral of the failure rate. As the integration interval becomes much larger than the rate at which transitions between states occur, the integrated failure rate converges first to a Gaussian distributed random variable, and for longer integration intervals, to a Wiener process.

This description has been completed by an approximation where the failure rate is not a two state process, but a Gaussian process resulting from averaging the real failure rate over a short period of time. In the case of digital computers this is just an approximation, but in other systems, a Gaussian process may be the actual failure rate.

It has been then shown how the statistics of the time to failure are completely determined by the expected value and variance of the process $\tilde{\Lambda}_t$, the integrated failure rate. Once these two moments are known, the failure process can be viewed as a nonhomogeneous Poisson process since all the functions on which the PDF depends are deterministic. The PDF and hazard function of the equivalent failure process are given in (3.103) and (3.105). Both depend only on the expected value and variance of the integrated failure rate since the system start time. These two moments depend in turn on the frequency with which requests arrive to the kernel, and on some statistics about how the requests are served. The following chapter specializes these results to two special cases of wide applicability.

Chapter 4
Specialization to systems under constant or periodic workload

In Chapter 3 the emphasis was on studying the more general properties of systems characterized by a new modeling methodology. This chapter derives some important properties for systems for which some more information is available. The system workload will be assumed now to be either constant or periodic. Nevertheless, it should be clear that periodicity or invariance is only average behavior. The actual failure rate is still assumed to be a stochastic process.

4.1. Case I - Constant workload

If the workload of the system is constant and the system is operating in steady state, it is reasonable to assume that the expected number of requests arriving to the kernel per unit time, $\nu(t)$ will be equal to a constant ν. In this case, the probability of observing n requests in a time interval $(t_2-t_1) = T$ is given by (3.11). Therefore, $\tilde{\lambda}_t$ becomes a stationary Gaussian process with mean

$$E[\tilde{\lambda}_t] = \frac{1}{W} E\left\{ \int_{t-w/2}^{t+w/2} \lambda_\tau \, d\tau \right\} \tag{4.1}$$

$$= \alpha + \beta \nu E[s_i] \tag{4.2}$$

$$= \alpha + \beta m \tag{4.3}$$

where m is the average fraction of time in kernel mode. Define then,

$$E[\tilde{\lambda}] \triangleq q \tag{4.4}$$

and

$$E\left[\tilde{\Lambda}_t \right] = q t \tag{4.5}$$

Since $\tilde{\lambda}_t$ is stationary, its autocorrelation function $R_{\tilde{\lambda}\tilde{\lambda}}(s,t)$ depends only on the difference $\tau = |s-t|$ and, from (3.102),

$$\mathrm{Var}\left[\tilde{\Lambda}_t\right] = 2\int_0^t (t-\tau)\left[R_{\tilde{\Lambda}\tilde{\Lambda}}(\tau) - q^2\right] d\tau \tag{4.6}$$

Let now

$$y_t = \tilde{\Lambda}_t - q \tag{4.7}$$

where y_t is a stationary, zero mean, Gaussian process, and

$$\mathrm{Var}\left[\tilde{\Lambda}_t\right] = 2\int_0^t (t-\tau) R_{yy}(\tau)\, d\tau \tag{4.8}$$

The equivalent nonhomogeneous failure process has then the following important properties :

Corollary 1: Let N_t be a failure process with failure rate process as defined in (3.10) with constant workload ($\nu(t) = \nu$) such that the failure rate can be approximated by

$$\tilde{\Lambda}_t = q + y_y \tag{4.9}$$

where y_t is a zero mean stationary Gaussian process. Define

$$W = \int_{-\infty}^{\infty} R_{yy}(\tau)\, d\tau \tag{4.10}$$

The statistics of the time to failure are then equal to the statistics of the time to failure of a *nonhomogeneous* Poisson process with hazard function h(t) such that

1. $h(0) = q$
2. $\lim_{t\to\infty} h(t) = q - \frac{W}{2}$, $W \geq 0$
3. If $R_{yy}(\tau)$ is nonnegative everywhere, then h(t) is nonincreasing.

PROOF : The hazard function of the equivalent process is given by (3.105). Substituting (4.6) and (4.5) in (3.105),

$$h(t) = q - \int_0^t R_{yy}(\tau)\, d\tau \tag{4.11}$$

and $h(0) = q$. For real processes, the autocorrelation is even and

$$\lim_{t\to\infty} h(t) = q - \frac{W}{2} \leq h(0) \tag{4.12}$$

Note that $W \geq 0$ because if $S_{yy}(\omega)$ is the power spectrum of y_y

$$S_{yy}(\omega) = \int_{-\infty}^{\infty} R_{yy}(\tau)\, e^{-j\omega\tau}\, d\tau \tag{4.13}$$

then $W = S_{yy}(0)$, which must be nonnegative for any phisical process.

Finally, if the autocorrelation function is nonnegative its integral is nondecreasing and h(t) as given in (4.11) must be *nonincreasing*. ∎

4.1.1. Examples

A complete family of distributions can now be obtained for the case of constant workload but different autocorrelation functions. The only restrictions are that being real processes, the autocorrelation functions must be even, positive definite, with a maximum at $\tau = 0$ and their integral over the real line must be nonnegative and bounded. The following examples illustrate some types of distributions that can be obtained under the assumption of constant workload.

4.1.1.1. Example 1. Exponentially decreasing hazard function - The doubly exponential distribution

If

$$R_{yy}(\tau) = \frac{W}{2}\ \beta\, e^{-\beta|\tau|} \tag{4.14}$$

then, the PDF of the time to failure is given by

$$P(t_f \leq \tau) = 1 - e^{-\left(q - \frac{W}{2}\right)\tau - \frac{W}{2\beta}\left[1 - e^{-\beta\tau}\right]} \tag{4.15}$$

and its hazard function is

$$h(\tau) = q - \frac{W}{2}\ \left[1 - e^{-\beta\tau}\right] \tag{4.16}$$

$$h(\infty) = q - \frac{W}{2} \tag{4.17}$$

Note that, as for any nonhomogeneous Poisson process, the hazard function is the derivative of the exponent in the PDF. In particular, if $q = \beta = 1$, $W = 2$,

$$P(t_f \leq \tau) = 1 - e^{e^{-\tau} - 1} \tag{4.18}$$

$$h(t) = e^{-t} \tag{4.19}$$

which is the doubly exponential distribution, one of the three possible (maximum) extreme value distributions (given that t_f must be nonnegative). Maximum extreme value distributions are obtained assuming that a system is formed by a collection of n identical modules operating in parallel. The system fails only when all the modules fail and the distribution of the time to system failure becomes the distribution of the maximum time to failure for the n modules. As n approaches infinity, the

distribution of the system time to failure converges in distribution either to an exponential, a Weibull distribution, or to the distribution given in (4.18) [Barlow 75].

More generally, if

$$R_{yy} = \sum_{i=1}^{k} \gamma_i e^{-\beta_i \tau} \tag{4.20}$$

then,

$$P(t_f \leq \tau) = 1 - e^{\left(q - \sum_{i=1}^{k} \frac{\gamma_i}{\beta_i}\right)\tau - \sum_{i=1}^{k} \frac{\gamma_i}{\beta_i^2}[1 - e^{-\beta_i \tau}]} \tag{4.21}$$

$$h(t) = q - \sum_{i=1}^{k} \frac{\gamma_i}{\beta_i} [1 - e^{-\beta_i t}] \tag{4.22}$$

This last distribution is commonly used in nuclear medicine to characterize the light pulses due to the absorption of gamma radiation emanating from radioactive tracers. The hazard function reflects physiological transport phenomena due to blood flow rate, metabolic exchange rates, and lung ventilation [Sheppard 62].

4.1.1.2. Example 2. The exponential distribution - white noise failure rate

If $\bar{\lambda}_t$ is white noise,

$$R_{yy} = \frac{W}{2} \delta(\tau) \tag{4.23}$$

where $\delta(t)$ is the Dirac delta function, then

$$P(t_f \leq \tau) = 1 - e^{\left(q - \frac{W}{2}\right)\tau} \tag{4.24}$$

$$h(t) = q - \frac{W}{2} \tag{4.25}$$

That is, the PDF degenerates into an exponential distribution. Note that its parameter is not equal to the mean failure rate (q) but to the mean failure rate minus the "power" of the process $\bar{\lambda}_t$, W/2.

4.1.1.3. Example 3. Pareto distribution

Assume that

$$R_{yy}(\tau) = \frac{\alpha \beta^2}{(\beta|\tau| + 1)^{2-\alpha}} \tag{4.26}$$

then

$$P(t_f \leq \tau) = 1 - e^{(q-\alpha\beta)\tau - \alpha \ln(\beta t + 1)} \tag{4.27}$$

$$= 1 - \frac{e^{-(q-\alpha\beta)t}}{(\beta t + 1)^{\alpha}} \tag{4.28}$$

In particular, if $m = \alpha\beta$,

$$P(t_f \leq \tau) = 1 - \frac{1}{(\beta t + 1)^{\alpha}} \tag{4.29}$$

$$h(t) = \frac{\alpha\beta}{\beta t + 1} \tag{4.30}$$

which is the Pareto distribution. The Pareto distribution is used to characterize clinical data relating to the probability of survival of individuals belonging to some populations. For instance, the Pareto distribution is postulated as the best distribution characterizing the probability that a patient waiting for a heart transplant (because of unavailable donors or other reasons) will die before receiving the heart transplant [Turnbull 74].

The choice of the Pareto distribution or distributions of the type (4.15) for analysis of survival data is common, and it is based mainly on heuristics. Hence, the present methodology justifies such choices whenever the actual failure rate can be approximated by a stationary Gaussian process with autocorrelation function given in (4.26).

4.1.1.4. Example 4. An intensity process with infinite energy - The Weibull distribution

Consider now the following sequence of stochastic processes. $y_t^{(n)}$ is a stationary Gaussian process with mean

$$q_n = \frac{\alpha\lambda}{(\lambda/n)^{1-\alpha}} \tag{4.31}$$

and autocorrelation function

$$R_{y_n y_n}(\tau) = \frac{(1-\alpha)\alpha\lambda^2}{(\lambda(t + 1/n))^{2-\alpha}} \tag{4.32}$$

where $\alpha<1$. Then,

$$P_n(t_f \leq \tau) = 1 - e^{-(\lambda(\tau + 1/n))^{\alpha} + (\lambda/n)^{\alpha}} \tag{4.33}$$

$$h_n(t) = \frac{\alpha\lambda}{(\lambda(t + 1/n))^{1-\alpha}} \tag{4.34}$$

Now let n go to ∞ and

$$P_n(t_f \leq \tau) \Rightarrow 1 - e^{-(\lambda\tau)^{\alpha}} \tag{4.35}$$

$$\lim_{n\to\infty} h_n(t) = \frac{\alpha\lambda}{(\lambda t)^{1-\alpha}} \tag{4.36}$$

which is the Weibull distribution. Note that as $n \to \infty$ both the mean value and the variance of $y_t^{(n)}$ also go to ∞. Hence, the process

$$y_t = \lim_{n\to\infty} y_t^{(n)} \tag{4.37}$$

is not physically realizable, since it has infinite energy. However, the fact that a limiting distribution exists for $P_n(t_f \leq \tau)$ indicates that the Weibull distribution may be the right choice for characterizing doubly stochastic Poisson processes with intensity processes that have very large mean and variance.

4.1.2. Discussion

These and other possible distributions are summarized in Table 4-1. The fact of considering the failure rate of a system to be a stationary Gaussian process is therefore a *unifying method for* obtaining a complete family of distributions commonly used in reliability theorey. Some more insight can be gained by careful examination of the similarities and differences between these distributions.

4.1.2.1. The distinctive property of white noise

The main difference between white noise and any other stationary Gaussian process is that of predictability. The best predictor (in a mean square sense) of y_t based on y_s, $s < t$, is $E[y_t|y_s = \xi]$. In general, for a stationary Gaussian process,

$$E[y_t|y_s = \xi] = \frac{R_{yy}(t-s)}{\sigma_y^2}\ \xi \tag{4.38}$$

([Wong 79], p. 64). However, if y_t is white noise,

$$E[y_t|y_s = \xi] = E[y_t] = 0 \qquad \text{if } s \neq t \tag{4.39}$$

White noise future values are totally unpredictable no matter how much information has been accumulated about its past behavior. On the other hand, for a nonwhite noise Gaussian processes there always exist constants $\alpha_k^{(n)}$ such that [Breiman 68]

$$E[y_{t_{n+1}}|y_{t_1} = \xi_1,\ldots,y_{t_n} = \xi_n] = \sum_{k=1}^{n} \alpha_k^{(n)}\ \xi_k \tag{4.40}$$

4.1.2.2. The rate of convergence to a Wiener process

As for the meaning of having a hazard function whose asymptotic value is smaller than the mean failure rate, note that the function

$$f(t) = \int_0^t R_{yy}(\tau)\, d\tau \tag{4.41}$$

is in fact the rate at which Λ_t converges to a Wiener process. In effect, note that

$$h(\infty) = q - \frac{W}{2} \tag{4.42}$$

is the hazard function that would be obtained if y_t were white noise and Λ_t were a Wiener process.

4.1.2.3. A different but equivalent conceptual framework

It is interesting to note how some of the distributions given in Table 4-1 can be obtained in a completely different way. Assume that the PDF of the time to failure is exponentially distributed with parameter λ, but that λ is a random variable. That is, once the system is started λ is chosen at random from a known distribution and remains constant until the system fails. Every time that the system is restarted, a new value of λ is randomly chosen. The PDF of the time to failures is in this case given by

$$P(t_f \leq \tau) = E\{P(t_f \leq \tau | \lambda)\} = E\left\{1 - e^{-\lambda\tau}\right\} \tag{4.43}$$

where the expectation is taken with respect the statistics of λ. It was first derived by [Harris 68] that, for instance, if λ is Gamma distributed,

$$p_\lambda(x) = \frac{\alpha}{\Gamma(\beta)} (\alpha x)^{\beta-1} e^{-\alpha x} \tag{4.44}$$

then $P(t_f \leq \tau)$ becomes the Pareto distribution. Similarly, other PDFs can be derived by assuming different distributions for λ.

If the failure process is a renewal process, the following three types of systems have identical statistics:

- Systems for which the failure process is a doubly stochastic Poisson process, $\lambda = q + y_t$, and $R_{yy}(\tau)$ leads to an equivalent hazard function h(t).
- Systems with a random hazard function λ such that $p_\lambda(x)$ leads to the same equivalent hazard function h(t).
- Systems for which the failure process is a nonhomogeneous Poisson process with hazard function h(t).

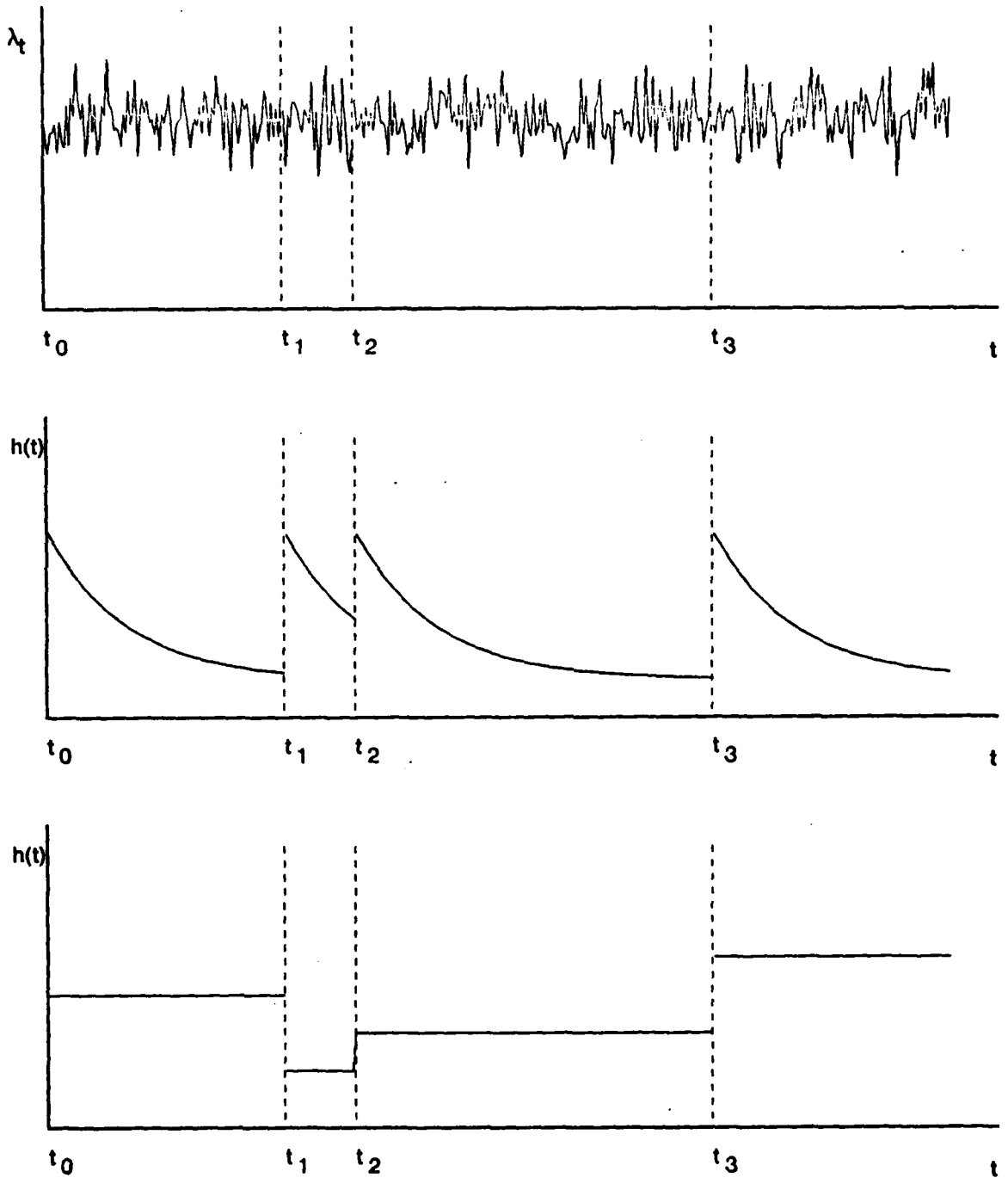


Figure 4-1: Three possible failure rates, all leading to the same statistics

Figure 4-1 illustrates the three types of systems. Which of the above three conceptual frameworks is more appropiate to work with will have to be decided usually after practical considerations. Probabilistically, the three types of systems are indistinguishable.

4.2. Case II - Periodic workload (the Cyclostationary process)

Assume now that $\nu(t)$ is a periodic function of t, that is

$$\nu(t) = \nu(t+T) \tag{4.45}$$

and

$$\tilde{\lambda}_t = \alpha + \frac{1}{W}\beta E[s_i] R_{[t-W/2,t+W/2]} + \frac{1}{W}\beta S_{R_{[t-W/2,t+W/2]}} \tag{4.46}$$

If $\nu(t)$ varies slowly enough such that it can be considered constant in any time interval $[t,t+w]$,

$$E[\tilde{\lambda}_t] = \alpha + \beta\nu(t)E[s_i] \tag{4.47}$$

$$\triangleq q(t) \tag{4.48}$$

where q(t) is also periodic with period T, and

$$E[\Lambda_{[t_1,t_2]}] = \int_{t_1}^{t_2} q(t)\,dt \tag{4.49}$$

Also,

$$R_{\tilde{\lambda}\tilde{\lambda}}(s,t) = E[\tilde{\lambda}_s\tilde{\lambda}_t]$$

$$= q(s)q(t) + \frac{\beta^2}{w^2} E\left[S_{R_{[t-W/2,t+W/2]}} S_{R_{[s-W/2,t+W/2]}} \right] \tag{4.50}$$

where

$$E\left[S_{R_{[t-W/2,t+W/2]}} S_{R_{[s-W/2,t+W/2]}} \right] = \sum_{n=0}^{\infty} P(R_{[s,t]} = n) \sum_{i=1}^{w\nu(t)} \sum_{j=i+n}^{i+n+w\nu(s)} E[x_i x_j] \tag{4.51}$$

Note that $q(t) = q(t+T)$ and that $R_{\tilde{\lambda}\tilde{\lambda}}(s,t) = R_{\tilde{\lambda}\tilde{\lambda}}(s+T,t+T)$. Thus, $\tilde{\lambda}$ is a cyclostationary process. As it has been remarked by [Gardner 78], if $N_{[t_1,t_2]}$ is a doubly stochastic Poisson process with cyclostationary intensity process, $N_{[t_1,t_2]}$ is itself cyclostationary, that is

$$E\left[N_{[t_1,t_2]}\right] = E\left[N_{[t_1+T,t_2+T]}\right] \tag{4.52}$$

(the remark in [Gardner 78] is for the more general concept of processes which are almost cyclostationary in the wide sense). The fact that $N_{[t_1,t_2]}$ is itself cyclostationary explains the data reported by [Butner 80], where the number of system failures as a function of time of day reflects the

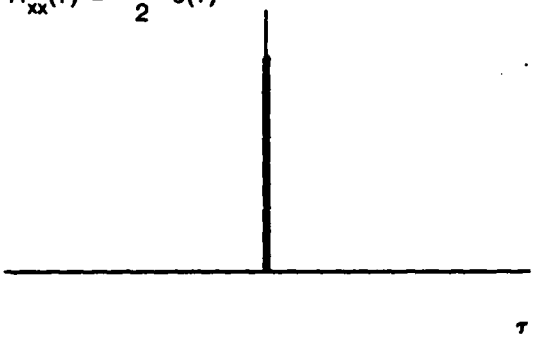
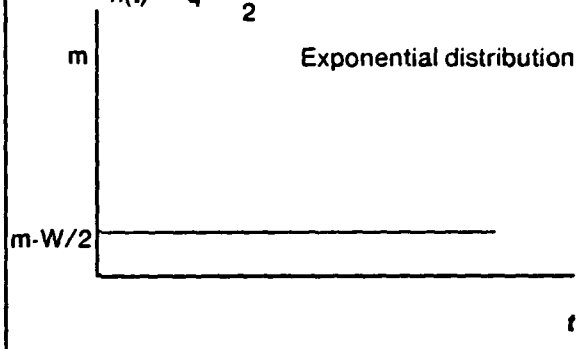
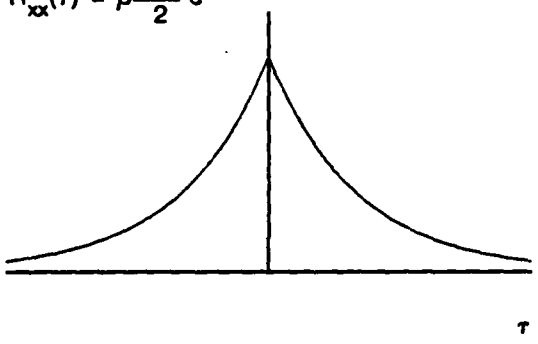
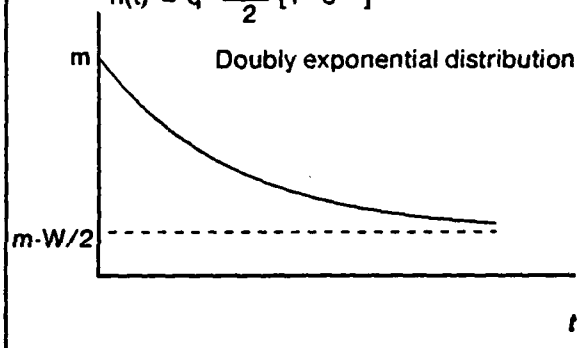
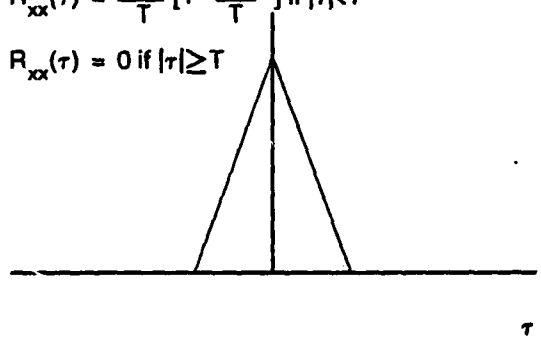
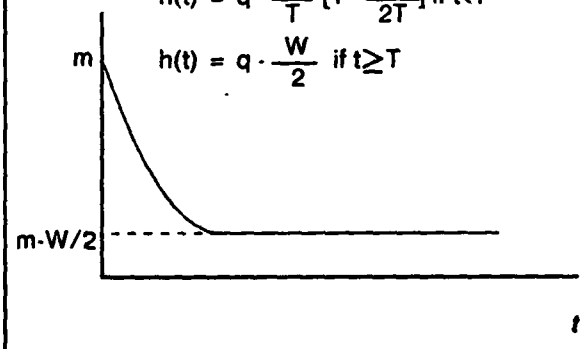
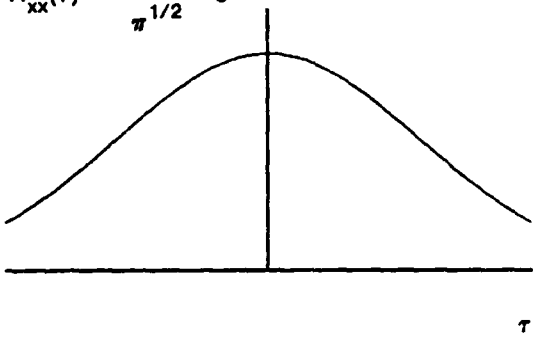
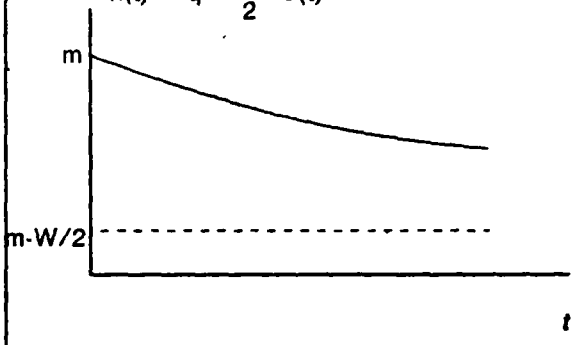
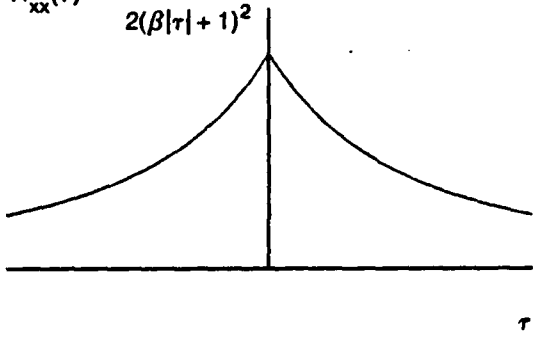
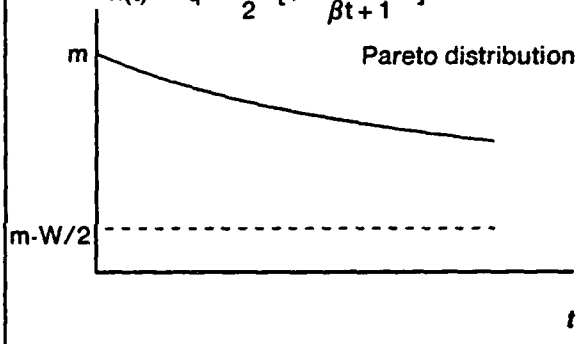
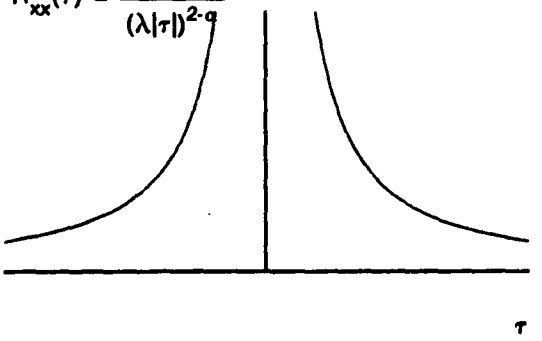
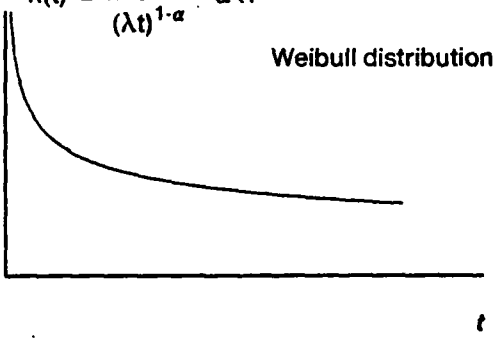
Autocorrelation function	Hazard function
$R_{xx}(\tau) = \frac{W}{2}\,\delta(\tau)$ 	$h(t) = q \cdot \frac{W}{2}$ 
$R_{xx}(\tau) = \beta\frac{W}{2}\,e^{-\beta\lvert\tau\rvert}$ 	$h(t) = q \cdot \frac{W}{2}\,[1 - e^{-\beta t}]$ 
$R_{xx}(\tau) = \frac{W}{T}\,[1 - \frac{\lvert\tau\rvert}{T}]$ if $\lvert\tau\rvert < T$ $R_{xx}(\tau) = 0$ if $\lvert\tau\rvert \geq T$ 	$h(t) = q \cdot \frac{Wt}{T}\,[1 - \frac{t}{2T}]$ if $t < T$ $h(t) = q \cdot \frac{W}{2}$ if $t \geq T$ 

Table 4-1: Examples of different autocorrelation functions and the corresponding hazard functions of the equivalent failure process

Autocorrelation function	Hazard function
$R_{xx}(\tau) = \frac{\alpha W}{\pi^{1/2}} e^{-\alpha t^2}$ 	$h(t) = q \cdot \frac{W}{2} \Phi(t)$ 
$R_{xx}(\tau) = \frac{W\beta}{2(\beta\lvert\tau\rvert + 1)^2}$ 	$h(t) = q \cdot \frac{W}{2} [1 - \frac{1}{\beta t + 1}]$ Pareto distribution 
$R_{xx}(\tau) = \frac{\alpha(1-\alpha)\lambda}{(\lambda\lvert\tau\rvert)^{2-\alpha}}$ 	$h(t) = \frac{\alpha\lambda}{(\lambda t)^{1-\alpha}}$ $\alpha<1$ Weibull distribution 

average workload variations over a one day period. This is a consequence of having a cyclostationary intensity process and does not imply a strictly periodic failure rate.

Define now,

$$y_t = \tilde{\lambda}_t - q(t) \tag{4.53}$$

and note that

$$R_{yy}(s,t) = \sum_{n=0}^{\infty} P(R_{[s,t]} = n) \sum_{i=1}^{w\nu(t)} \sum_{j=i+n}^{i+n+w\nu(s)} E[x_i x_j] \tag{4.54}$$

By the properties of an α-mixing sequence, $E[x_i x_j]$ approaches zero as the difference $|i-j|$ increases. Thus, $R_{yy}(s,t)$ should approach zero as the difference $|s-t|$ increases. Further, note that

$$R_{yy}(t,t) = \sigma_y^2(t) = \nu(t) \frac{\beta^2}{W} \left[E[s_i]^2 + \sigma^2 \right] \tag{4.55}$$

Thus, $R_{yy}(s,t)$ can be expressed as

$$R_{yy}(s,t) = \sigma_y(s)\sigma_y(t)\, \eta(|t-s|) \tag{4.56}$$

where $\eta(x)$ is a function of x such that $\eta(0) = 1$ and $\eta(\infty) = 0$. Therefore,

$$\mathrm{Var}\left[\tilde{\Lambda}_{[t_1,t_2]}\right] = \int_{t_1}^{t_2} \int_{t_1}^{t_2} R_{yy}(s,t)\, ds\, dt \tag{4.57}$$

$$= \int_{t_1}^{t_2} \int_{t_1}^{t_2} \sigma_y(s)\sigma_y(t)\eta(|s-t|)\, ds\, dt \tag{4.58}$$

which after some algebraic manipulations can be shown to be equal to

$$\mathrm{Var}\left[\Lambda_{[t_1,t_2]}\right] = \Sigma_y(t_1,t_2) \int_{t_1}^{t_2} \sigma_y(t)\eta(|t_2-t|)\, dt - \sigma_y(t_1) \int_{t_1}^{t_2} \Sigma_y(t_1,t)\eta(t)\, dt \tag{4.59}$$

where

$$\Sigma_y(t_1,t) = \int_{t_1}^{t} \sigma_y(\tau)\, d\tau \tag{4.60}$$

4.2.1. Two important properties of the cyclostationary Poisson process

Although not as simple as the case of stationarity intensity, closed form expressions for the hazard function and PDF of the time to failure for cyclostationary failure processes can be obtained. The only restriction is that now the hazard function and PDF are conditioned to the starting time value.

Corollary 2: Let $N_{[t_1,t_2]}$ be a doubly stochastic Poisson process with cylostationary intensity process λ_t such that

$$\lambda_t = q(t) + y_t \tag{4.61}$$

$$R_{yy}(s,t) = \sigma_y(s)\sigma_y(t)\eta(|s-t|) \tag{4.62}$$

$$q(t) = q(t+T) \qquad \sigma_y(t) = \sigma_y(t+T) \tag{4.63}$$

The hazard function of the equivalent nonhomogeneous Poisson process given that the system is started at time t_s is

$$h(t|t_s) = q(t) \cdot \sigma_y(t) \int_{t_s}^{t} \sigma_y(\tau)\, \eta(|t-\tau|)\, d\tau \tag{4.64}$$

and its conditional PDF is

$$P(t_f \le \tau | t_s) = 1 - \exp\Big\{ \int_{t_s}^{\tau} q(t)\, dt - \Sigma_y(t_s,\tau) \int_{t_s}^{\tau} \sigma_y(t)\, \eta(|\tau-t|)\, dt + \sigma_y(\tau) \int_{t_s}^{\tau} \Sigma_y(t_s,t)\, \eta(t)\, dt \Big\} \tag{4.65}$$

PROOF: From (4.58), (3.101), and (3.103), after (substantial) algebraic manipulations. ∎

The above hazard function and PDF are conditioned to the starting time value t_s. To obtain the unconditional functions, the following expectations should be computed.

$$h(t) = \int_0^{\infty} p_{t_s}(u)\, h(t|t_s = u)\, du \tag{4.66}$$

$$P(t_f \le \tau) = \int_0^{\infty} p_{t_s}(u)\, P(t_f \le \tau | t_s = u)\, du \tag{4.67}$$

where $p_{t_s}(u)$ is the pdf of the starting time. The following theorem gives the value of this distribution. Its simplicity has very important practical implications.

Theorem 3: Let $N_{[t_1,t_2]}$ be a doubly stochastic Poisson process with intensity given in (4.61) through (4.63). Assume that the system is observed for n consecutive cycles and let $P_n(t_s \le \tau)$ denote the PDF of the system start time during these n cycles. Then

$$P_n(t_s \le \tau) \Rightarrow P(t_s \le \tau) = \frac{1}{\int_0^T q(s)\, ds} \int_0^{\tau} q(s)\, ds \qquad 0 \le \tau \le T \tag{4.68}$$

or, equivalently,

$$p_{t_s}(u) = \lim_{n\to\infty} p^n_{t_s}(u) = \frac{q(u)}{\int_0^T q(s)\,ds} \tag{4.69}$$

if

$$\lim_{n\to\infty} \frac{1}{n^2} \int_0^{nT}\int_0^{nT} \sigma_y(s)\sigma_y(t)\,\eta(|s-t|)\,ds\,dt = 0 \tag{4.70}$$

PROOF: Assume that n_f failures have been observed in n cycles. Then,

$$p^n_{t_{f_i}}(u|\lambda_t, 0\le t\le nT; N_{[0,nT]} = n_f) = \frac{\lambda_u}{\int_0^{nT} \lambda_y\,dy} \qquad i = 1,\ldots,nf \tag{4.71}$$

Since $t_{f_i} = t_{s_{i+1}}$, the above density is also the pdf of t_{s_i}, $i = 2,\ldots,n_f+1$. Further, note that the t_{f_i} are mutually independent and that the above pdf is the same for any value of $n_f > 0$. Thus,

$$p^n_{t_{s_i}}(u|y_t, 0\le t\le nT) = \frac{q(u)}{\int_0^{nT} q(s)\,ds + \int_0^{nT} y_s\,ds} + \frac{y_u}{\int_0^{nT} q(s)\,ds + \int_0^{nT} y_s\,ds} \tag{4.72}$$

Since $q(t) = q(t+T)$

$$p^n_{t_{s_i}}(u|y_t, 0\le t\le nT) = \frac{n\,q(u')}{n\int_0^{T} q(s)\,ds + \int_0^{nT} y_s\,ds} + \frac{y_u}{n\int_0^{T} q(s)\,ds + \int_0^{nT} y_s\,ds} \tag{4.73}$$

where

$$u' = u - \lfloor u/nT \rfloor \tag{4.74}$$

Hence,

$$p_{t_{s_i}}(u) = \lim_{n\to\infty} E\left\{\frac{q(u')}{\int_0^T q(s)\,ds + (1/n)\int_0^{nT} y_s\,ds}\right\} + E\left\{\frac{y_u/n}{\int_0^T q(s)\,ds + (1/n)\int_0^{nT} y_s\,ds}\right\} \tag{4.75}$$

Now, if

$$s_n = (1/n)\int_0^{nT} y_s\,ds \tag{4.76}$$

s_n is a Gaussian random variable with

$$E[s_n] = 0 \tag{4.77}$$

$$\mathrm{Var}[s_n] = \frac{1}{n^2}\int_0^{nT}\int_0^{nT} \sigma_y(s)\sigma_y(t)\eta(|s-t|)\,ds\,dt \tag{4.78}$$

Therefore, if (4.70) holds, $s_n = 0$ with probability one as $n\to\infty$ and

$$\lim_{n\to\infty} E\left\{\frac{q(u')}{\int_0^T q(s)\,ds + (1/n)\int_0^{nT} y_s\,ds}\right\} = \frac{q(u')}{\int_0^T q(s)\,ds} \tag{4.79}$$

As for the second term in (4.75), note that

$$\frac{1}{n}E[y_n s_n] = \frac{1}{n^2}\int_0^{nT} R_{yy}(|v-u|)\,dv \tag{4.80}$$

Since y_n is a physical process, the integral in (4.80) remains bounded as the upper limit goes to ∞. Therefore y_n and s_n become uncorrelated and independent (both are Gaussian random variables) as $n\to\infty$. Thus,

$$\lim_{n\to\infty} E\left\{\frac{\dot{y}_u/n}{\int_0^T q(s)\,ds + (1/n)\int_0^{nT} y_s\,ds}\right\} = 0 \tag{4.81}$$

which completes the proof. ∎

4.3. Summary

The analysis of systems under constant average workload has lead to a complete family of distributions commonly used in reliability theory. The distinctive property between different distributions is the autocorrelation function of the intensity process. The fact that all distributions have limiting hazard function values smaller than the average failure rate is of particular importance. The limiting hazard function value is the value that would be obtained if the failure rate were white noise. The rate of convergence of the integrated failure rate to a Wiener process has been shown to be the integral of the autocorrelation function.

It is important to note that the rate of convergence to a Wiener process is one of the parameters characterizing the reliability of the system under study. Consider two identical systems, A and B, such that the failure rate of system A is white noise, while the failure rate of system B is some other Gaussian process. Although both systems can do the same amount of work in the same time (in the sense that the expected value of the integrated failure rate is the same) system A is more reliable than system B. The integral of the failure rate for system A *is* a Wiener process no matter how short the

integration interval is. Therefore, system A reaches the asymptotic (minimum) value of its hazard function instantly. This point will be elaborated later on in the thesis and will be illustrated with numerical examples.

The analysis of systems under periodic workload has not resulted in so concise results. However, an important property of cyclostationary failure processes is that, for systems operating after many cycles, the distribution of the system failure time over one cycle converges to the periodic component of the failure rate. This fact will lead in Chapter 7 to the establishement of cost functions on which cost-benefit analysis of fault-tolerance can be based.

Throught the Chapter it has been assumed that the failure rate can be approximated by a deterministic function of time plus a zero mean Gaussian process,

$$\lambda_t = q(t) + y_t \tag{4.82}$$

However, from chapter 3 it is known that

$$\lambda_t = p_u + \frac{1}{W}(p_k - p_u + p_s)(m(t) + x_t) \tag{4.83}$$

where $m(t) + x_t$ is the fraction of time in kernel mode in the interval $[t-W/2, t+W/2]$ and p_u, p_k, p_s are the coefficients establishing the sensitivity of the system to different failures depending on the system state. Therefore, λ_t can be rewritten as

$$\lambda_t = \kappa_1 + \kappa_2(m(t) + x_t) \tag{4.84}$$

Hence, the failure process can be characterized by a doubly stochastic Poisson process with intensity

$$\lambda_t = f(m(t), x_t, \vec{\kappa}) \tag{4.85}$$

where f() is an arbitrary function and $\vec{\kappa}$ is a vector of coefficients. In all cases, the PFD of the time to failure can be expressed as

$$P(t_i \leq \tau | t_s) = 1 - e^{-\int_s^{\tau} h(t)\,dt} \tag{4.86}$$

where h(t) is the hazard function of the equivalent nonhomogeneous process.

Chapter 5
Failure process analysis of a real system

5.1. System characteristics and measuring tools

In order to verify that the model described in Chapter 3 leads to a better fit to failure processes than previous work, an experiment was designed. Data was acquired for both the failure processes and the load of a general purpose time sharing system. The system chosen was the CMU-A, a PDP-10 used by the Computer Science Department at Carnegie-Mellon University as its main general purpose computational system. The system consists of a KL-10 processor, one megaword of memory, eight disk drives totaling 1600 megabytes of online storage and two magnetic tape drives. The system runs a slightly modified version of the standard TOPS-10 operating system [Bell 78].

The software packages used to instrument the experiment are illustrated in Figure 5-1. Information about failures is obtained from an online error log file maintained by a system program, which records the information produced by different error formatting routines. Entries are made to this file for each hardware error detected in the system, for system reloads, for disk performance statistics, and so on [Digital 78]. The error log is later processed by SEADS, a FORTRAN package which lists the times of detection of errors associated with a particular resource. In order to obtain accurate information about the use of the system, a special SAIL program, SYSMON, was written that periodically samples the values of 30 system parameters. The files generated by SYSMON are later processed by another SAIL package, READSY, which computes the periodic component and autocorrelation function of the utilization function of a particular system resource. The information generated by SEADS and READSY is then processed by an APL package (POWELL) which estimates the maximum likelihood parameters of the pdf of the time to failure of a particular resource. Finally, in a separate SAIL package, C2TST, the values predicted by the cyclostationary model and other models described in Section 4 are compared with the information stored in the error log according to a χ^2 goodness-of-fit test.

The value of the accumulated time spent in kernel mode is obtained by executing a Monitor Call and includes the time spent in clock queue processing, short command processing, swapping and scheduling decisions, and software context switching [Digital 77]. This value does not include Monitor Call execution nor I/O interrupt times. The sampled value is not exactly the time that the system is executing in kernel mode, but it is close enough for our purposes.

5.2. Model parameterization

According to the results presented in Chapter 3, the failure process of a Time Sharing computing system can be characterized by a doubly stochastic Poisson process with intensity process

$$\lambda_t = f(m(t), x_t, \vec{\kappa}) \tag{5.1}$$

where

$$k_t = m(t) + x_t \tag{5.2}$$

is the average fraction of time spent in kernel mode in an interval of duration W centered at time t and $\vec{\kappa}$ is a vector of parameters. In order to parameterize our model, the values of λ_t must be sampled from real systems, and from these samples m(t) and the autocorrelation function of x_t must be estimated. Further, methods for estimating the maximum likelihood values of $\vec{\kappa}$ must also be provided.

5.2.1. Sampling the intensity process

The operating system automatically measures the cumulative time spent in kernel mode. That means that the value of

$$K_t = \int_{t_s}^{t} k_\tau \, d\tau \tag{5.3}$$

can be easily sampled. If the value of K_t are sampled at times $\{t_{n-W/2}, t_{n+W/2}, t_{n+1-W/2}, \ldots\}$ samples of the observable intensity process are immediately available as

$$k_{t_n} = K_{t_{n+W/2}} - K_{t_{n-W/2}} \tag{5.4}$$

where $t_n = t_s + n\Delta t$ and t_s is the system start time.

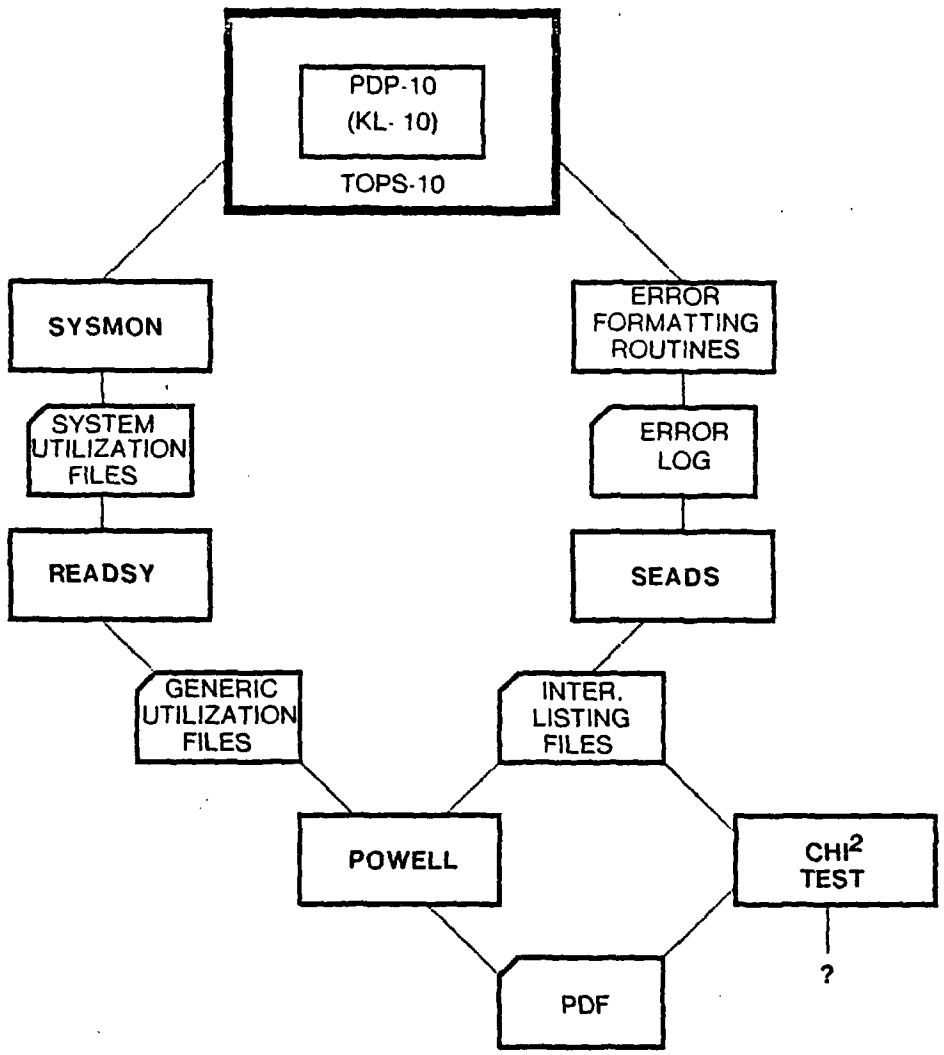


Figure 5-1: Software packages used in the validation of the cyclostationary modeling methodology.

5.2.2. Estimating the deterministic component

The expected value of k_t, m(t), is a deterministic function of time with period T = 24 hours. Thus, m(t) can easily be estimated from the samples k_{t_n}. If data has been collected for N days, let

$$m'(t_n) = \frac{1}{N} \sum_{j=1}^{N} k_{t_n + jT} \tag{5.5}$$

m(t) will be then approximated by a finite Fourier series expansion, that is,

$$m(t) = m + \sum_{n=1}^{N} c_n \sin(n\omega t + \varphi_n) \tag{5.6}$$

where the following constants have been used

$$\omega = \frac{2\pi}{T} \qquad m = \frac{1}{T} \int_0^T m'(t)\, dt \tag{5.7}$$

$$c_n = (a_n^2 + b_n^2)^{1/2} \qquad \varphi_n = \arctan \frac{a_n}{b_n} \tag{5.8}$$

$$a_n = \frac{2}{T} \int_0^T m'(t) \cos(n\omega t)\, dt \qquad b_n = \frac{2}{T} \int_0^T m'(t) \sin(n\omega t)\, dt \tag{5.9}$$

5.2.3. Autocorrelation function estimation

Given an ergodic and stationary process z_t, the problem is to estimate the function

$$R_{zz}(\tau) = \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} z_{t+\tau} z_t\, dt \tag{5.10}$$

For a finite record of observed values z_{t_n}, n = 1,...,N,, the autocorrelation function is usually estimated using the expression

$$R_{zz}(n) = \frac{1}{N} \sum_{i=1}^{N-n} z_{t_{i+n}} z_{t_i} \tag{5.11}$$

This estimate is intuitive except for the factor 1/n. Since N-n terms are summed, it seems that 1/(N-n) would be more exact. In fact (5.11) is a biased estimator of the real autocorrelation function. However, its expected error is smaller than the expected error that would be obtained using the (unbiased) estimator with factor 1/(N-n) [Jenkins 68].

In the cases presented in this thesis the values of z_{t_n} are not directly observable. For the sampling of the fraction of time in Kernel mode, what is measured is the average fraction of time in Kernel mode during a period of duration W. The measured values are not the values of z_{t_n}, but the values of the process

$$\tilde{z}_{t_n} = \int_{t_n - W/2}^{t_n + W/2} z_t \, dt \tag{5.12}$$

As will be shown in the following sections, in the two cases studied in this thesis the autocorrelation function suggests an approximation of the form

$$R_{zz}(t) = \alpha_1 e^{-\beta_1 |t|} + \alpha_2 e^{-\beta_2 |t|} \tag{5.13}$$

The problem is then to estimate the values of the α_i, β_i from the observed values of $\tilde{z}_{t_n}$. If

$$R_{\tilde{z}\tilde{z}}(t) = \alpha'_1 e^{-\beta_1 |t|} + \alpha'_2 e^{-\beta_2 |t|} \tag{5.14}$$

it is easy to show that

$$R_{zz}(t) = \alpha_1 e^{-\beta_1 |t|} + \alpha_2 e^{-\beta_2 |t|} \tag{5.15}$$

where

$$\alpha_i = \frac{\beta_i^2 \alpha_i'}{2[\cosh(\beta_i W) - 1]} \tag{5.16}$$

The problem is then to estimate the values of the α'_i, β_i using (5.11) and the observed values of $\tilde{z}_{t_n}$, and use (5.16) to obtain the values of α_i of the autocorrelation function of z_t.

Unfortunately it will not always be possible to follow this procedure. The accuracy of the estimated autocorrelation function is limited basically by two factors : the sampling frequency and the length of the available record, N. Although many techniques exist for power spectrum estimation that take into account these two factors [Oppenheim 75] (the power spectrum is the Fourier transform of the autocorrelation function), no techniques are available for correcting the estimates of the autocorrelation function itself. If the sampling frequency is comparable to the bandwidth of the power spectrum, the power spectrum estimate may be poor due to aliasing. Under these conditions, the estimate of the autocorrelation function given by (5.11) may take negative values.

5.2.4. Maximum likelihood estimation of model coefficients

The general problem of parameter estimation for doubly stochastic Poisson processes can be stated as follows. Let $\{N(t);t>t_0\}$ be a doubly stochastic Poisson counting process with intensity $\lambda(t,x_t,\vec{\kappa})$, where x_t is a stochastic process and $\vec{\kappa} = (\kappa_1,\kappa_2,.....,\kappa_m)$ is a vector of unknown parameters. The occurrence density function that a given realization of the process has a failure at time t_f if it has been started at time t_s is, given by

$$p(t_f|\vec{\kappa},x_\tau,t_s<\tau<t_f) = \lambda(t_f,x_{t_f},\vec{\kappa})\, e^{-\int_{t_s}^{t_f} \lambda(\tau,x_\tau,\vec{\kappa})\, d\tau} \tag{5.17}$$

If n failures are observed at times $t_{f_1},.....,t_{f_n}$ with associated starting times $t_{s_1},.....,t_{s_n}$, the probability density function of observing such set of events is

$$p^{(n)}(t_{f_1},....,t_{f_n}|\vec{\kappa},x_\tau,t_{s_i}<\tau<t_{f_i},i=1,...,n) = \prod_{i=1}^{n} P(t_{s_i})\,\lambda(t_{f_i},x_{t_{f_i}},\vec{\kappa})\, e^{-\int_{t_{s_i}}^{t_{f_i}} \lambda(\tau,x_\tau,\vec{\kappa})\, d\tau} \tag{5.18}$$

where $P(t_{s_i})$ is the a priori probability that the system is started at time t_{s_i}. Taking the expectation with respect the statistics of x_t we can obtain,

$$p^{(n)}(t_{f_1},....,t_{f_n}|\vec{\kappa},t_{s_i},....,t_{s_n}) = E\Big\{ \prod_{i=1}^{n} P(t_{s_i})\,\lambda(t_{f_i},x(t_{f_i}),\vec{\kappa})\, e^{-\int_{t_{s_i}}^{t_{f_i}} \lambda(\tau,x_\tau,\vec{\kappa})\, d\tau} \Big\} \tag{5.19}$$

The maximum likelihood estimate $\vec{\kappa}' = (\kappa'_1,\kappa'_2,.....,\kappa'_m)$ of $\vec{\kappa}$ in terms of a particular realization of the process is by definition the value of $\vec{\kappa}$ that maximizes the above density function [Melsa 78]. That is, $p^{(n)}(t_{f_1},.....,t_{f_n}|\vec{\kappa},t_{s_i},i=1,.....,n)$ will be maximum for $\vec{\kappa} = \vec{\kappa}'$. As it has been shown in Chapter 3, the pdf of the time to failure can be written as

$$p(t) = h(t_f,\vec{\kappa})\, e^{-\int_{t_s}^{t_f} h(\tau,\vec{\kappa})\, d\tau} \tag{5.20}$$

the function to be maximized is then

$$p^{(n)}(t_{f_1},.....,t_{f_n}) = \prod_{i=1}^{n} P(t_{s_i})\, h(t_{f_i},\vec{\kappa})\, e^{-\int_{t_{f_i}}^{t_{s_i}} h(\tau,\vec{\kappa})\, d\tau} \tag{5.21}$$

Note that this problem is equivalent to minimizing the function

$$L(\vec{\kappa}) = \Big(\sum_{i=1}^{n} \int_{t_{s_i}}^{t_{f_i}} h(\tau,\vec{\kappa})\, d\tau\Big) \cdot \Big(\sum_{i=1}^{n} \ln[h(t_{f_i},\vec{\kappa})]\Big) \tag{5.22}$$

subject to the constraints

$$h(t_{f_i},\vec{\kappa})>0 \qquad i=1,.....,n \tag{5.23}$$

Since closed form expressions for the components of $\vec{\kappa}$ at the minimum are not generaly available, this is a typical nonlinear programming problem, subject to nonlinear inequality constraints. Since this problem will have to be solved every time that the failure process of a resource has to be modeled for a real system, particular care has been taken in finding an efficient procedure for the location of minimums of functions of the type (5.22). The algorithm used is a slightly modified version of a variable metric algorithm proposed by [Powell 78]. The original Powell algorithm occasionally requires the evaluation of the objective function outside the constraints and has been modified such that the maximum step size at each iteration never leads to a point outside the constraints. The algorithm has been implemented as an APL package that requires the definitions of the objective function, gradient, and constraints. Several objective functions corresponding to different distributions were given in [Castillo 80b].

5.2.5. Error correction

The last practical consideration to be treated in this section deals with the approximation of k_t as a Gaussian random variable. If k_t is a Gaussian random variable with mean m(t) and variance σ^2, there is a finite probability that $k_t<0$

$$P(k_t<0) = \frac{1}{(2\pi)^{1/2}\sigma}\int_{-\infty}^{-m(t)} e^{-x^2/2\sigma^2}\,dx \tag{5.24}$$

However, since k_t is the sum of positive random variables, it can never actually be negative. If k_t can never be smaller that a nonnegative value k_{min}, a better approximation of k_t is

$$k_t = \begin{cases} m(t)+x_t & \text{if } m(t)+x_t>k_{min} \\ k_{min} & \text{otherwise} \end{cases} \tag{5.25}$$

Define then

$$k_t = m(t) + x_t - x_t^c \tag{5.26}$$

where

$$x_t^c = \begin{cases} m(t)+x_t-k_{min} & \text{if } m(t)+x_t<k_{min} \\ 0 & \text{if } m(t)+x_t\geq k_{min} \end{cases} \tag{5.27}$$

The problem is therefore to evaluate the expectation

$$E\left\{e^{\int_{s}^{\tau}(x_t - x_t^c)\,dt}\right\} \tag{5.28}$$

Note that the integral

$$\int_{t_s}^{\tau} x_t^c dt \tag{5.29}$$

is equal to the excess area under the peaks of x_t below a threshold k_{min}-m(t) (see Figure 5-2).

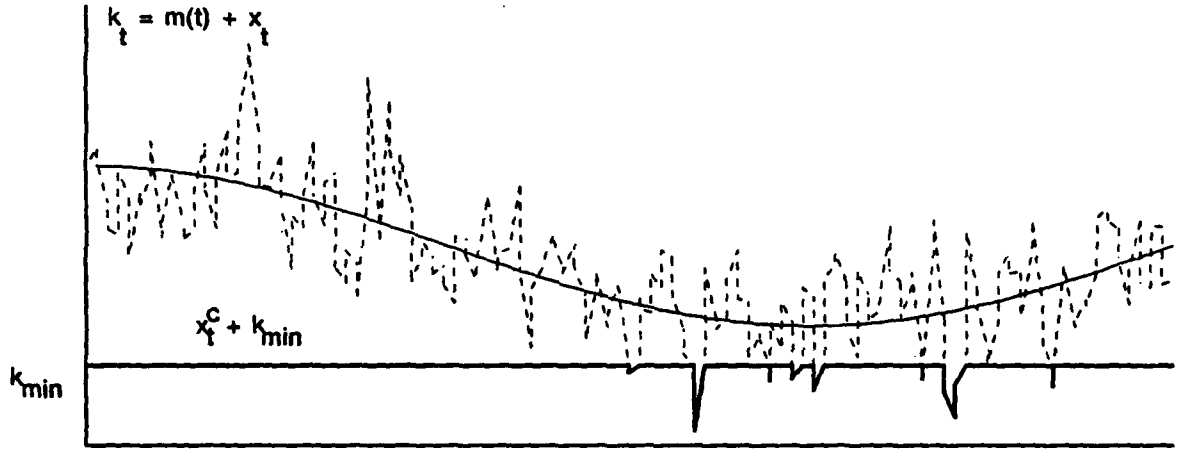


Figure 5-2: Relationship between x_t and x_t^c.

It is shown in [Stratonovich 67] that if the duration of the peaks is much smaller than the time between peaks, the occurrence times of peaks above a threshold c(t) can be approximated by a Poisson process with intensity

$$\eta(t) = \frac{(R_2)^{1/2}}{2}\, e^{-c^2(t)/2\sigma^2} \tag{5.30}$$

where

$$R_2 = -\left[\frac{\partial^2 R_{xx}(\tau)}{\partial \tau^2}\right]_{\tau=0} \tag{5.31}$$

and σ^2 is the variance of x_t. Since the presence of peaks can be characterized as a Poisson process, the excess areas under the peaks can be viewed as a marked Poisson process (see [Snyder 75], Ch. 7) with nondenumerable mark space. Let s_i denote the area under the *i*-th peak in the interval [*s,t*]. s_i will be the mark associated with the i-th point (peak) in [s,t].

The statistics of the sum of the areas under the peaks are equivalent to the statistics of the mark-accumulator process,

$$\int_{t_s}^{\tau} x_t^c = \sum_{i=0}^{N_{[t_s,\tau]}} s_i \tag{5.32}$$

where $N_{[t_s,\tau]}$ is the number of peaks in $[t_s,\tau]$. The above expectation can therefore be rewritten as ([Snyder 75],p131)

$$E\left\{e^{\int_{ts}^{\tau} x_t^c dt}\right\} = P(N_{[t_s,\tau]}=0) + \sum_{n=1}^{\infty} P(N_{[t_s,\tau]}=n)\, E\left\{e^{\sum_{i=1}^{n} s_i} \middle| N_{[t_s,\tau]}=n\right\} \tag{5.33}$$

Note that

$$E\left\{e^{\sum_{i=1}^{n} s_i} \middle| N_{[t_s,\tau]}=n\right\} = E\left\{E\left[e^{\sum_{i=1}^{n} s_i} \middle| N_{[t_s,\tau]}=n; t_1, t_2, \ldots, t_{N_{[t_s,\tau]}}\right] \middle| N_{[t_s,\tau]}=n\right\} \tag{5.34}$$

Given $N_{[t_s,\tau]}=n$, $t_1,\ldots,t_{N_{[t_s,\tau]}}$ are a collection of independent, identically distributed random variables ([Snyder 75], p. 65), the common distribution being

$$p_{t_i}(x) = \frac{\eta(x)}{\int_{t_s}^{\tau} \eta(t)\, dt} \qquad t_s \le x \le \tau \tag{5.35}$$

Therefore, if the areas under different peaks are mutually independent,

$$E\left[e^{\sum_{i=1}^{n} s_i} \middle| N_{[t_s,\tau]}=n; t_1, \ldots, t_{N_{[t_s,\tau]}}\right] = \left[E\left\{e^{s_1}\right\}\right]^n \tag{5.36}$$

and, if the i-th peak occurs at time t_i ,

$$E\left\{e^{s_1}\right\} = \int_{t_s}^{\tau} p_{t_i}(x)\, E[e^{s_{t_i}} | t_i = x]\, dx \tag{5.37}$$

$$= \int_{t_s}^{\tau} p_{t_i}(x) \int_0^{\infty} p_{s_\tau | t_i}(\chi | t_i = x)\, e^{\chi}\, d\chi\, dx \tag{5.38}$$

where

$$p_{s_\tau}(\chi) = \frac{1}{3}\left[\frac{3c^2(\tau)}{3\sigma^3}\right]^{2/3} \chi^{-1/3}\, e^{-\frac{1}{2}\left[\frac{3c^2(\tau)}{2\sigma^3} R_2^{1/2} \chi\right]^{1/3}} \tag{5.39}$$

and

$$c(\tau) = k_{min} \cdot m(t) \tag{5.40}$$

If $s_{t_i} \ll 1$ for all i, the following approximation can be made

$$E\left\{e^{s_1}\right\} = 1 + \int_{t_s}^{\tau} p_{t_i}(x) E[s_{t_i} | t_i = x] dx \tag{5.41}$$

and

$$E[s_{t_i} | t_i = x] = \int_0^\infty x p_{s_x}(x)\, dx \tag{5.42}$$

$$= \frac{\sigma^3}{c^2(x)} \left(\frac{2\pi}{R_2} \right)^{1/2} \tag{5.43}$$

Define then

$$E\left\{ e^{s_i} \right\} = 1 + \zeta(t_s, \tau) \tag{5.44}$$

where

$$\zeta(t_s, \tau) = \frac{\sigma^3 (2\pi / R_2)^{1/2}}{\int_{t_s}^{\tau} e^{-c^2(\tau)/2\sigma^2} d\tau} \int_{t_s}^{\tau} \frac{e^{-c^2(\tau)/\sigma^2}}{c^2(\tau)} \, d\tau \tag{5.45}$$

Therefore,

$$E\left\{ e^{\int_s^\tau x_t^c \, dt} \right\} = P(N_{[t_s,\tau]} = 0) + \sum_{n=1}^{\infty} P(N_{[t_s,\tau]} = n) [1 + \zeta(t_s, \tau)]^n \tag{5.46}$$

and, since $N_{[t_s,\tau]}$ is a Poisson process with intensity $\eta(t)$,

$$E\left\{ e^{\int_s^\tau x_t^c \, dt} \right\} = \sum_{n=0}^{\infty} \frac{\left[[1 + \zeta(t_s, \tau)] \int_{t_s}^{\tau} \eta(t)\, dt \right]^n}{n!} e^{\int_s^\tau \eta(t)\, dt} \tag{5.47}$$

$$= e^{\zeta(t_s,\tau) \int_s^\tau \eta(t)\, dt} \tag{5.48}$$

Note that if $c(\tau) = c$ independent of τ and x_t is stationary,

$$E\left\{ e^{\int_s^\tau x_t^c dt} \right\} = e^{\rho(c)(\tau - t_s)}$$

where

$$\rho(c) = \frac{\sigma^3 (\pi/2)^{1/2}}{c^2}$$

5.3. Characterization of the time to System Failure

In this section the modeling methodology presented in Chapter 3 will be applied to characterize the reliability of the system described in Section 4.1. All the necessary techniques to parameterize the model have been given in Section 4.2. An exact characterization will be developed in Section 4.3.1, where the periodicity of the workload is taken into account. In Section 4.3.2 the model is simplified giving the characterization that would result from assuming a constant workload, and therefore a stationary intensity process. Figures 5-3 through 5-7 summarize the behavior observed of the CMU-A.

Figure 5-3 gives the actual values of the average fraction of time in kernel mode, k_t, averaged over one second and sampled every five minutes for five consecutive weekdays. The periodicity of the mean is clear from this figure. As a further indication that k_t can be approximated by a cyclostationary process, Figure 5-4 shows the estimated autocorrelation function of k_t, $R_{kk}(\tau)$, according to equation (5.11). $R_{kk}(\tau)$ is obviously periodic with a period of 24 hours. The estimated autocorrelation function was obtained from a record of samples k_{t_n} covering 60 days of normal system operation.

Figure 5-5 shows the estimated *average* fraction of time in kernel mode m'(t), and its Fourier series expansion, m(t), obtained as described in Section 5.2.2. Figure 5-6 shows the histogram of system failures as a function of time of day. To study the properties of the stochastic component, x_t, a plot of the variance $\sigma_x^2(t)$ as a function of time of day is given in Figure 5-7. The variance is about two orders of magnitude smaller than the mean m(t). Therefore, the error correction term given in Section 5.2.5. should be very small. Note also that the variance is approximately constant over a one day period. The peak between 9:00 and 10:00 is probably due to the fact that the system is started between those times after daily preventive maintenance . Therefore, x_t can be approximated by a stationary Gaussian process (although the results given in Chapter 4 predict a periodic variance, this periodicity is not noticeable here).

In summary, the *instantaneous* fraction of time in kernel mode can be approximated by

$$k_t = m(t) + x_t \tag{5.49}$$

where m(t) is periodic, x_t stationary, and k_t cyclostationary.

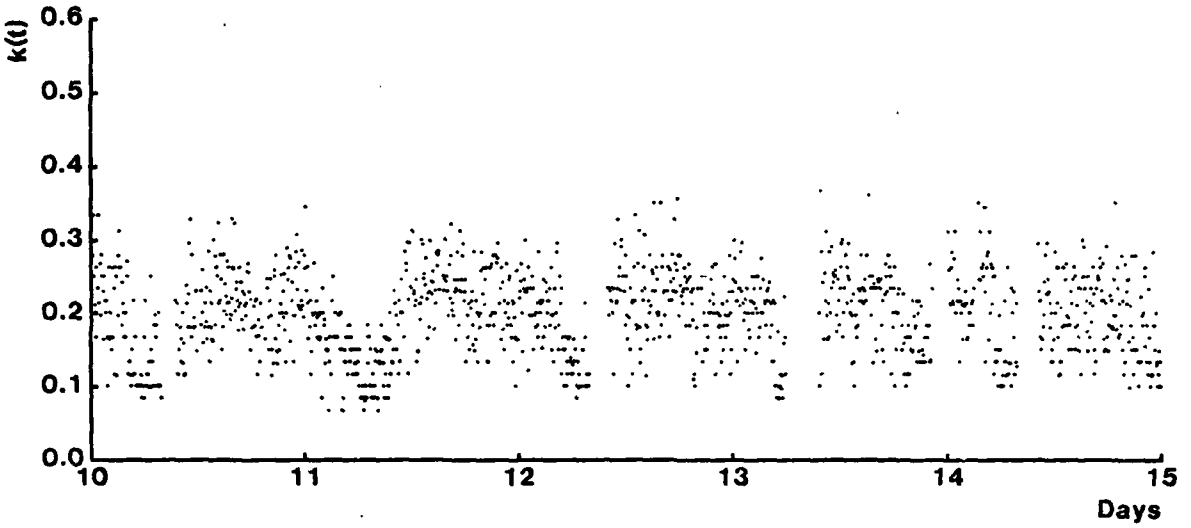


Figure 5-3: Fraction of time in kernel mode for five consecutive weekdays

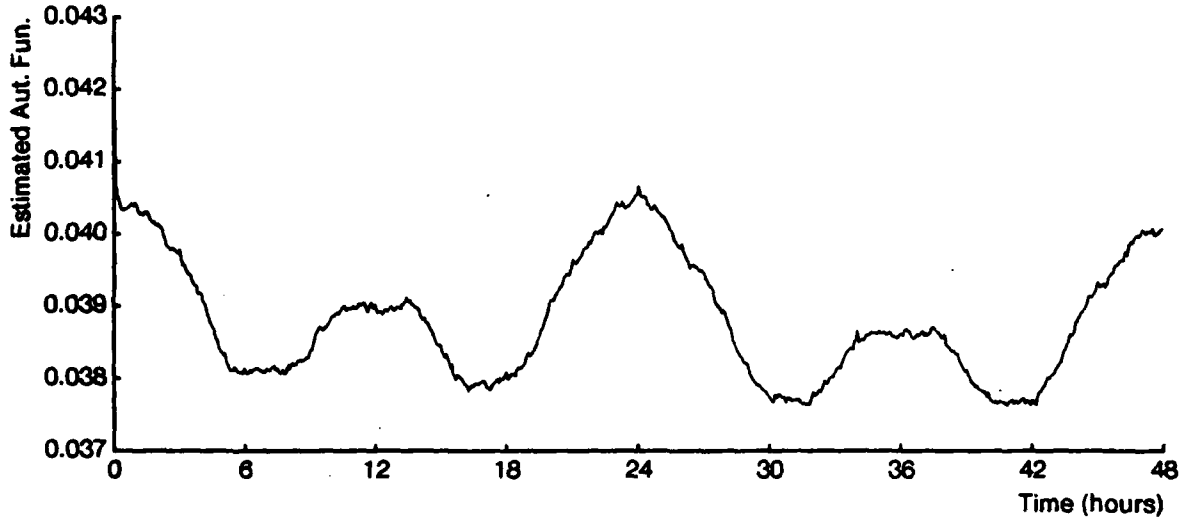


Figure 5-4: Autocorrelation function of k_t

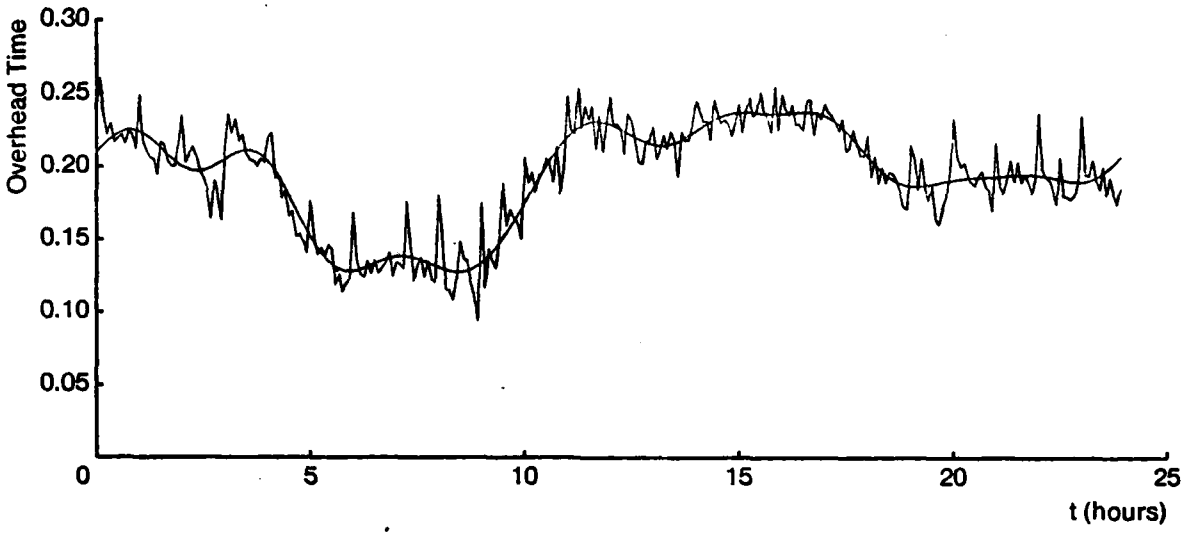


Figure 5-5: Fraction of time in kernel mode averaged over a one day period

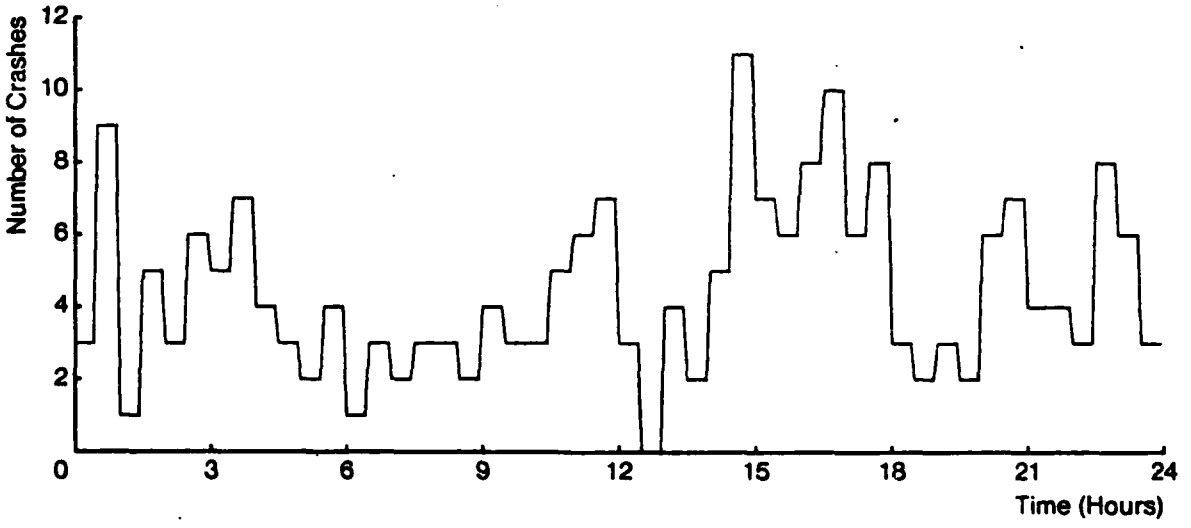


Figure 5-6: Number of system failures as a function of time of day

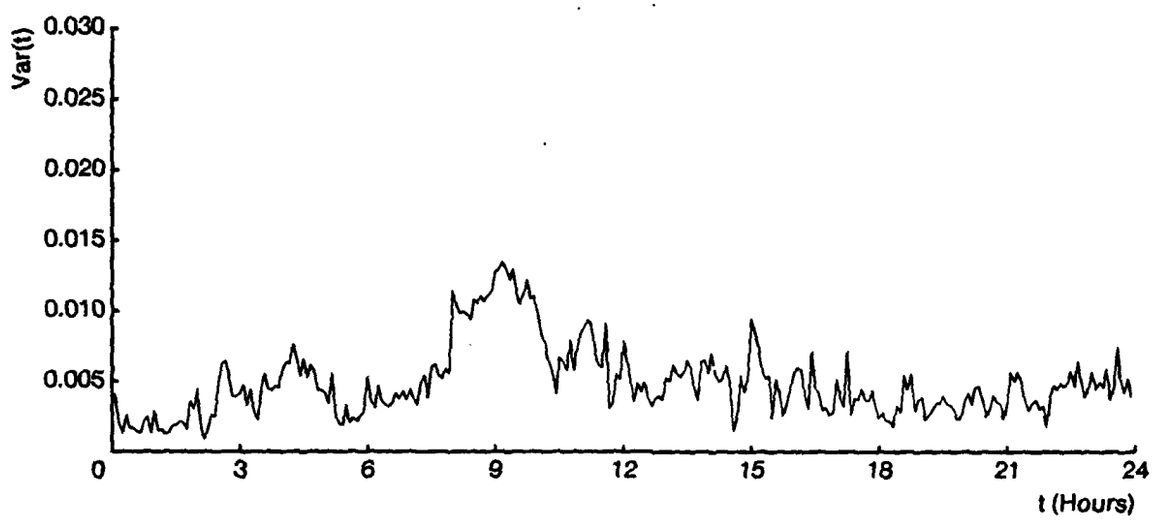


Figure 5-7: Variance of x_t averaged over a one day period

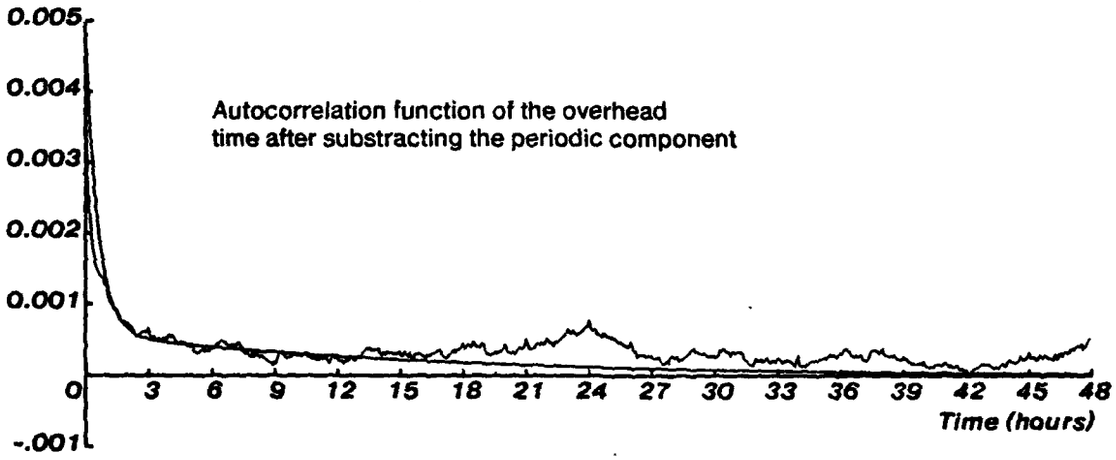


Figure 5-8: Estimated and approximated autocorrelation function of the process x_t

5.3.1. The cyclostationary model

With the notation developed in Chapter 3, recall that if $\bar{\lambda}$ is the failure rate of a Time Sharing system,

$$\bar{\lambda}_t = p_u + (p_k - p_u + p_s)k_t \tag{5.50}$$

where k_t is the fraction of time in kernel mode averaged in an interval $[t-W/2, t+W/2]$ and p_u, p_k, and p_s are different parameters reflecting the sensitivity of the system to transient faults and software faults. To remember the meaning of each parameter, $\bar{\lambda}_t$ will be rewritten as

$$\bar{\lambda}_t = c_{hw} + (s_{hw} + s_{sw})k_t \tag{5.51}$$

c_{hw} is a constant (workload independent) failure rate due to hardware transient faults, s_{hw} is a sensitivity coefficient relating the kernel usage with the (workload dependent) failure rate due to transients, and s_{sw} is an analogous sensitivity coefficient for the failure rate due to software faults.

The autocorrelation function of the process x_t is shown in Figure 5-8 suggesting that an approximation of the form

$$R_{xx}(t) = \alpha_1 e^{-\beta_1|t|} + \alpha_2 e^{-\beta_2|t|} \tag{5.52}$$

is appropriate to describe it. Using the results given in Chapter 3, the PDF of the time to failure conditioned to a starting time t_s is given by

$$P(t \le t_f | t_s) = 1 - e^{-(c_{hw} - \sigma_1 - \sigma_2)(t_f - t_s) - s_{sy}\int_{t_s}^{t_f} m(t)\,dt - \frac{\sigma_1}{\beta_1}[1-e^{-\beta_1(t_f - t_s)}] - \frac{\sigma_2}{\beta_2}[1-e^{-\beta_2(t_f - t_s)}]} \tag{5.53}$$

where the following constants have been defined

$$s_{sy} = s_{sw} + s_{hw} \tag{5.54}$$

$$\sigma_1 = s_{sy}^2 \frac{\alpha_1}{\beta_1} \tag{5.55}$$

$$\sigma_2 = s_{sy}^2 \frac{\alpha_2}{\beta_2} \tag{5.56}$$

and the unconditional PDF is given by

$$P_{sy}(t_f \leq \tau) = 1 - \varphi_{sy}(\tau)\, e^{-(\alpha_{sy}-\sigma_{sy1}-\sigma_{sy2})\tau - \frac{\sigma_{sy1}}{\beta_1}[1-e^{-\beta_1\tau}] - \frac{\sigma_{sy2}}{\beta_2}[1-e^{-\beta_2\tau}]} \tag{5.57}$$

where

$$\alpha_{sy} = (s_{hw} + s_{sw})\,\overline{m} + c_{hw} - \rho(s_{hw} + s_{sw}, k_{min}) \tag{5.58}$$

$$\sigma_{sy1} = (s_{sw} + s_{hw})^2 \frac{\alpha_1}{\beta_1} \tag{5.59}$$

$$\sigma_{sy2} = (s_{sw} + s_{hw})^2 \frac{\alpha_2}{\beta_2} \tag{5.60}$$

$$\varphi_{sy}(t) = \frac{1}{\int_0^T m(t)\,dt} \int_0^T m(\tau)\, e^{-\int_\tau^{\tau+t} m(s)\,ds}\, d\tau \tag{5.61}$$

5.3.2. The stationary approximation

Some approximations can be made leading to simpler expressions for the PDF of the time to failure. In particular, the periodicity of the workload will be neglected and the system failure rate will be assumed to be

$$\lambda_t = c + s\,k_t \tag{5.62}$$

where

$$k_t = m + x_t \tag{5.63}$$

and x_t is a stationary Gaussian process with autocorrelation function

$$R_{xx}(\tau) = \overline{\sigma}^2\, \eta(|\tau|) \tag{5.64}$$

If this autocorrelation function is of the form

$$R_{xx}(\tau) = \alpha_1 e^{-\beta_1|\tau|} + \alpha_2 e^{-\beta_2|\tau|} \tag{5.65}$$

using the results of section 3.5.1.2 the following expressions are obtained for the PDF and hazard function of the time to system failure

$$P(t_f \leq \tau) = 1 - \exp\left\{ -\left(c + sm \cdot s^2 \frac{\alpha_1}{\beta_1} \cdot s^2 \frac{\alpha_2}{\beta_2} \right) \tau \right.$$

$$\left. \cdot s^2 \frac{\alpha_1}{\beta_1^2}\left[1-e^{-\beta_1 \tau}\right] \cdot \frac{\alpha_2}{\beta_1^2}\left[1-e^{-\beta_2 \tau}\right] \right\} \tag{5.66}$$

$$h(t) = c + sm \cdot s^2 \frac{\alpha_1}{\beta_1} \left[1 - e^{-\beta_1 \tau}\right] \cdot s^2 \frac{\alpha_2}{\beta_1} \left[1 - e^{-\beta_1 \tau}\right] \tag{5.67}$$

Table 5-1 shows the maximum likelihood values for both the Stationary and Cyclostationary approximations computed from a history of 243 system failures (crashes) from December of 1979 to May 1980. After performing a χ^2 goodness-of-fit test between the predicted and observed distribution of failures, both approximations gave levels of confidence larger than 0.05, suggesting the acceptance of both distributions as good characterizations of the PDF of the time to failure.

Figure 5-9 shows the hazard function of the equivalent nonhomogeneous process Poisson process for both the Cyclostationary and Stationary approximations. The periodic component of the failure rate has been dampened so much that only the exponentially decreasing effect can be observed, and the Cyclostationary and Stationary hazard functions are undistinguishable.

A further approximation can be made. If the autocorrelation function is simplified to a single exponential,

$$R_{xx}(\tau) = \alpha e^{-\beta|\tau|} \tag{5.68}$$

then

$$P(t_f \leq \tau) = 1 - e^{-(c + sm \cdot s^2 \frac{\alpha}{\beta})\tau \cdot s^2 \frac{\alpha}{\beta^2}[1-e^{-\beta\tau}]} \tag{5.69}$$

$$h(t) = c + sm \cdot s^2 \frac{\alpha}{\beta} \left[1 - e^{-\beta\tau}\right] \tag{5.70}$$

5.3.3. A further refinement of the cyclostationary model

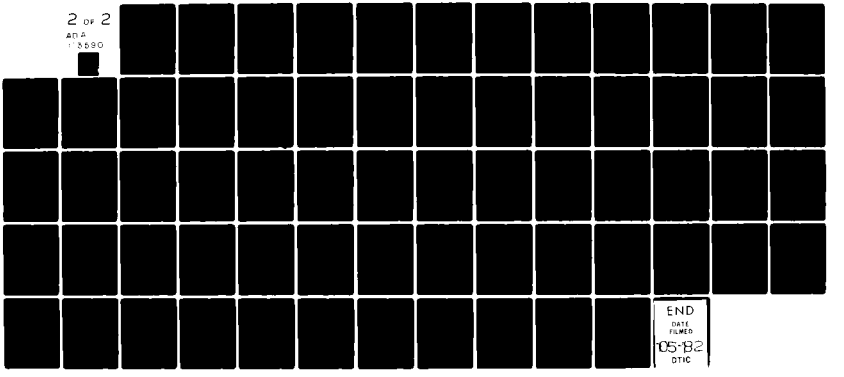
Equation (5.51) implies that, while the system is in kernel mode, the probability of observing a failure due to software on a time interval Δt is

$$p_{sw}(\Delta t) = s_{sw} \Delta t \tag{5.71}$$

which is a constant independent of the state of the system. This can hardly be a reasonable

AD-A113 590 CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER --ETC F/G 9/2
A COMPATIBLE HARDWARE/SOFTWARE RELIABILITY PREDICTION MODEL.(U)
JUL 81 X CASTILLO, T D SMITH DAS660-80-C-0057
UNCLASSIFIED CMU-CS-81-138 NL

2 OF 2
AD A
113590



END
DATE FILMED
05-82
DTIC

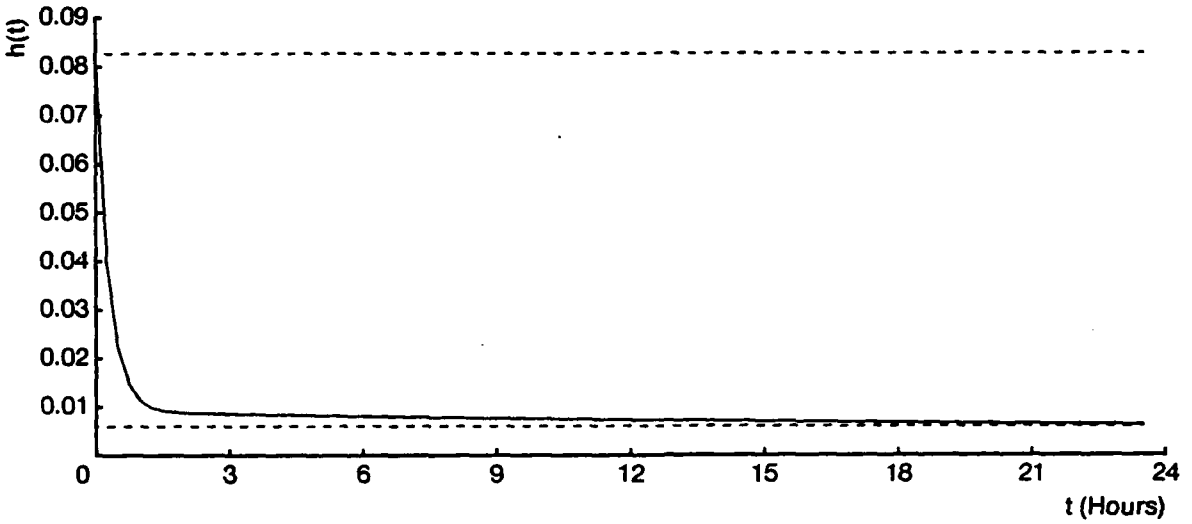


Figure 5-9: Hazard function of the equivalent nonhomogeneous Poisson process describing the system failure process in both the Cyclostationary and Stationary forms. The two dashed lines indicate the values of the hazard function at zero and infinity.

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Cyclostat.	$s_{sy} = 2.23$ $c_{hw} = 0.082$	18	15.89	27.869	0.6
Stationary	$\alpha_{sy} = 0.08$ $\sigma_{sy1} = 0.073$ $\sigma_{sy2} = 0.0041$ $\beta_1 = 0.28$ $\beta_2 = 0.0039$	14	15.89	23.68	0.31

Table 5-1: Results of applying a χ^2 goodness-of-fit test for the Cyclostationary and Stationary models for system failures (crashes). Both models give levels of confidence larger than 0.05, therefore confirming their validity as accurate system characterization tools

approximation. If, as described in Chapter 2, software unreliability is mainly due to persistent errors

deriving from oversimplifying the complexity of the data to be processed, s_{sw} should not be a constant. In effect, the instantaneous probability of observing a failure due to a software fault should increase when the system is processing data with large complexity and should decrease when processing simple data. Thus, s_{sw} should be a time varying function $s_{sw}(t)$ whose instantaneous value will depend on the average complexity of the data to be processed at time t. The problem is therefore how to characterize data complexity since, if c(t) is a suitable descriptor of the complexity of the data at time t,

$$s_{sw}(t) = f(c(t))$$

where f(x) is a nondecreasing function of x. The most easily measurable descriptor of data complexity is the *average* time spent in kernel mode, m(t). In effect, a large value of m(t) indicates a highly loaded system, impliying therefore a large number of decisions to be taken by the kernel per unit time and continuous updating of its data structures. A small value of m(t) indicates a lightly loaded system, with relatively static and half empty data structures. Note that the *instantaneous* value of the fraction of time in kernel mode, k_t is *not* a good descriptor of data complexity because the fact that for a second the kernel has been executed a very short period of time is not meaningful (perhaps a large number of jobs were just waiting for I/O completion).

$s_{sw}(t)$ will therefore be assumed to be a nondecreasing function of m(t). Again for simplicity a linear relationship will be assumed such that

$$s_{sw}(t) = s_{sw_1} m(t) + s_{sw_2} \tag{5.72}$$

$$\lambda_t = c_{hw} + [s_{sy} + s_{sw_1} m(t)][m(t) + x_t] \tag{5.73}$$

where $s_{hw} + s_{sw_2}$ has been noted s_{sy}. If

$$q(t) = s_{sy} + s_{sw_1} m(t) \tag{5.74}$$

then

$$\lambda_t = c_{hw} + q(t)m(t) + q(t)x_t \tag{5.75}$$

$$= m'(t,\vec{\kappa}) + x'_t(\vec{\kappa}) \tag{5.76}$$

where $\kappa_1 = c_{hw}$, $\kappa_2 = s_{sy}$, $\kappa_3 = s_{sw_1}$, and $\vec{\kappa} = (\kappa_1,\kappa_2,\kappa_3)$. Using the results given in Chapter 3, the PDF and hazard function of the time to failure are easily obtained. Juts note that

$$R_{x'x'}(s,t) = E[x'_s x'_t] \tag{5.77}$$

$$= q(s)q(t)E[x_s x_t] \tag{5.78}$$

$$= \sigma'_x(s)\sigma'_x(t)\eta(|s-t|) \tag{5.79}$$

where

$$\sigma'_x(t) = q(t)\sigma_x \tag{5.80}$$

Therefore the PDF of the time to system failure is given by (4.67) evaluated by substituting m(t) by m'(t) and $\sigma_x(t)$ by $\sigma'_x(t)$ in (4.64) and (4.65). Given a set of n observations of system starting and failing time $\{[t_{s_i},t_{f_i}]; i = 1,...,n\}$, according to section 5.2.4, the maximum likelihood estimators of $\vec{\kappa}$ is the value of $\vec{\kappa}$ which minimizes the function

$$L(\vec{\kappa}) = \sum_{i=1}^{n} H(t_{s_i},t_{f_i},\vec{\kappa}) - \sum_{i=1}^{n} \ln[h(t_{s_i},t_{f_i},\vec{\kappa})] \tag{5.81}$$

subject to the constraints

$$h(t_{s_i},t_{f_i},\vec{\kappa}) > 0 \; i = 1,...,n \tag{5.82}$$

$$\kappa_i \geq 0 \; i = 1,...,3 \tag{5.83}$$

The values of $h(t_{s_i},t_{f_i},\vec{\kappa})$ and $H(t_{s_i},t_{f_i},\vec{\kappa})$ can be obtained from the results presented in Section 3.6.

$$h(t_{s_i},t_{f_i},\vec{\kappa}) = m'(t_{f_i}) - \sigma'_x(t_{f_i}) \int_{t_{s_i}}^{t_{f_i}} \sigma'_x(\tau)\, \eta(|t_{f_i}-\tau|)\, d\tau \tag{5.84}$$

$$H(t_{s_i},t_{f_i},\vec{\kappa}) = \int_{t_{s_i}}^{t_{f_i}} m'(t)\, dt - \Sigma'_x(t_{s_i},t_{f_i}) \int_{t_{s_i}}^{t_{f_i}} \sigma'_x(t)\, \eta(|t_{f_i}-t|)\, dt + \sigma'_x(t_{f_i}) \int_{t_{s_i}}^{t_{f_i}} \Sigma'_x(t_{s_i},t)\, \eta(t)\, dt \tag{5.85}$$

where

$$\Sigma'_x(a,b) = \int_a^b \sigma'_x(t)\, dt \tag{5.86}$$

and m', σ'_x, η, Σ'_x are functions of $\vec{\kappa}$.

The minimization of $l(\vec{\kappa})$ in (5.81) is a well defined non linear programming problem. However, the relationships between the affected variables are cumbersome. A simpler method to evaluate a good estimate of $\vec{\kappa}$ would be helpful.

5.3.4. A computational shortcut

In Section 3.6.2 it was shown how for a system under periodic workload the distribution of the system failure time (system start time) as a function of time of day should approach the average intensity, a linear function of the average workload. Thus, one would expect that for an observation interval sufficiently long, a histogram of the system failure time as a function of time of day should approach m'(t). Assume that such a histogram has been evaluated in C(t)

$$C(t) = n \qquad \text{if the number of failures in } [\, t/\Delta t, t/\Delta t + \Delta t \,] = n \tag{5.87}$$

Recall that the system failure rate can be expressed as

$$\lambda = f(m(t),\vec{\kappa}) + x_t(\vec{\kappa}) \tag{5.88}$$

Therefore, a possible estimate of $\vec{\kappa}$ is the value of $\vec{\kappa}$ which minimizes the norm

$$N(\vec{\kappa}) = \left|\left| C(t), f(m(t),\vec{\kappa}) \right|\right| \tag{5.89}$$

defined in a suitable functional space. In particular, if the norm chosen is $L^2_{[0,T]}$, the estimate of $\vec{\kappa}$ will be that value of $\vec{\kappa}$ which minimizes the function

$$N(\vec{\kappa}) = \int_0^T \left[\, C(t) - f(m(t),\vec{\kappa}) \,\right]^2 dt \tag{5.90}$$

Differentiation with respect to κ_i, the following system of equations is obtained

$$2\int_0^T C(t)\frac{\partial f(m(t),\vec{\kappa})}{\partial \kappa_i}\, dt = \int_0^T f(m(t),\vec{\kappa})\frac{\partial f(m(t),\vec{\kappa})}{\partial \kappa_i}\, dt \qquad i = 1,\ldots,n \tag{5.91}$$

where n is the number of components of $\vec{\kappa}$. In particular, if $f(m(t),\vec{\kappa})$ is a polynomial of order n-1 on m(t),

$$f(m(t),\vec{\kappa}) = \sum_{i=1}^{n} \kappa_i [m(t)]^{i-1} \tag{5.92}$$

the following system of n equations is obtained

$$\chi_j = \sum_{i=1}^{n} \kappa_i \mu_{i+j-1} \qquad j = 1,\ldots,n \tag{5.93}$$

where

$$\chi_j = \int_0^T C(t)[m(t)]^{j-1}\, dt \qquad j = 1,\ldots,n \tag{5.94}$$

$$\mu_i = \int_0^T [m(t)]^{i-1}\, dt \qquad i = 1,\ldots,2n-1 \tag{5.95}$$

If $f(m(t),\vec{\kappa})$ is of the form given in (5.73), a system of three equations is obtained with $\kappa_1 = c_{hw}$, $\kappa_2 = s_{sy}$, $\kappa_3 = s_{sw_1}$.

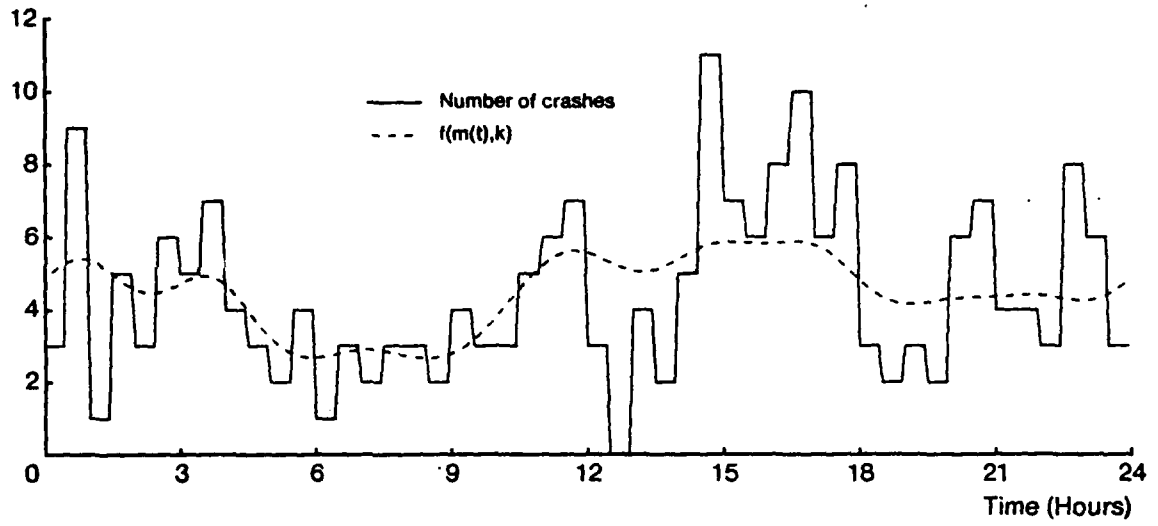


Figure 5-10: Periodic failure rate component compared with a real histogram of failures over a one day period.

The values of c_{hw}, s_{sy}, and s_{sw_2} were obtained from the histogram of failure data shown in 5-6 and m(t), the average fraction of time in kernel mode. Figure 5-10 shows the histogram of failures and the function $f(m(t),\vec{\kappa})$

$$f(m(t),\vec{\kappa}) = c_{hw} + (s_{hw} + s_{sw_1})\, m(t) + s_{sw_2}\, m^2(t) \tag{5.96}$$

This result will be used in Chapter 7 to evaluate the contribution of software to system unreliability.

5.4. Probability Distribution Function of the Time to Failure of a File System

The modeling methodology presented in Chapters 3 and 4 can be used to characterize the reliability of other systems or resources besides a complete Time Sharing system. As a final example (which will be also validated) the PDF of the time to failure of a file system will be evaluated.

For a file system, the reasoning is that errors can be detected only when accessing it. The assumptions are that all errors are hardware transients and that the instantaneous failure rate value is

$$\lambda_t^{dk} = f(m^{dk}(t), x_t^{dk}, \vec{\kappa}) \tag{5.97}$$

where

$$b_t = m^{dk}(t) + x_t^{dk} \tag{5.98}$$

is the number of blocks accessed in the interval $[t-W/2, t+W/2]$. Again, it is assumed that m^{dk} is periodic, x_t stationary and that

$$\lambda_t^{dk} = c_{dk} + s_{dk}[m^{dk}(t) + x_t^{dk}] \tag{5.99}$$

is cyclostationary.

Figure 5-11 shows the results of compiling five days of disk utilization samples into a single 24 hour period. Along with the estimated average, this figure shows the function $m^{dk}(t)$ obtained from a finite Fourier series expansion. After substracting from b_t the value of $m^{dk}(t)$, the sampled values of the process x_t^{dk} are available for estimation of its autocorrelation function.

The estimated autocorrelation of x_t^{dk} also suggests that an approximation of the form

$$R_{zz}(t) = \alpha_1 e^{-\beta_1|t|} + \alpha_2 e^{-\beta_2|t|} \tag{5.100}$$

would be appropriate to approximate the real autocorrelation function.

$$P_{dk}(t<\tau) = 1 - \phi_{dk}(\tau)\, e^{-(\alpha_{dk}-\sigma_{dk1}-\sigma_{dk2})t - \frac{\sigma_{dk1}}{\beta_1}[1-e^{-\beta_1 t}] - \frac{\sigma_{dk2}}{\beta_2}[1-e^{-\beta_2 t}]} \tag{5.101}$$

where the following constants and functions have been defined

$$\alpha_{dk} = s_{dk}\bar{m}_{dk} + c_{dk} - \rho_{dk}(s_{dk}, b_{min}) \tag{5.102}$$

$$\sigma_{dk1} = \frac{\alpha_1}{\beta_1} s_{dk}^2 \tag{5.103}$$

$$\sigma_{dk2} = \frac{\alpha_2}{\beta_2} s_{dk}^2 \tag{5.104}$$

$$\varphi_{dk}(t) = \frac{1}{\int_0^T m^{dk}(t)\, dt} \int_0^T e^{\int_\tau^{\tau+t} m^{dk}(s)\, ds}\, d\tau \tag{5.105}$$

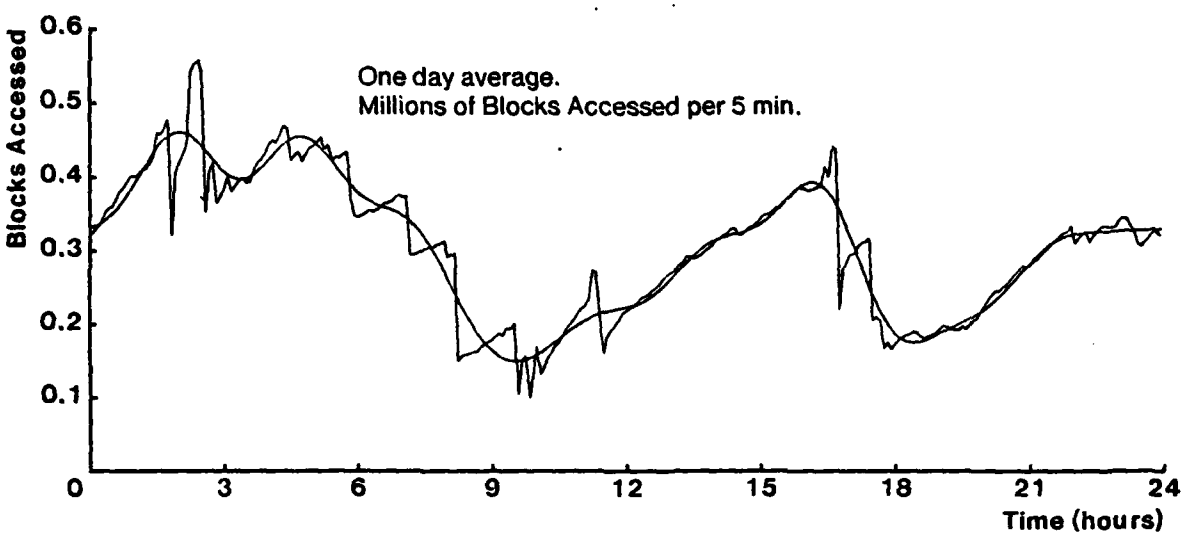


Figure 5-11: Estimated and approximated value of $m^{dk}(t)$

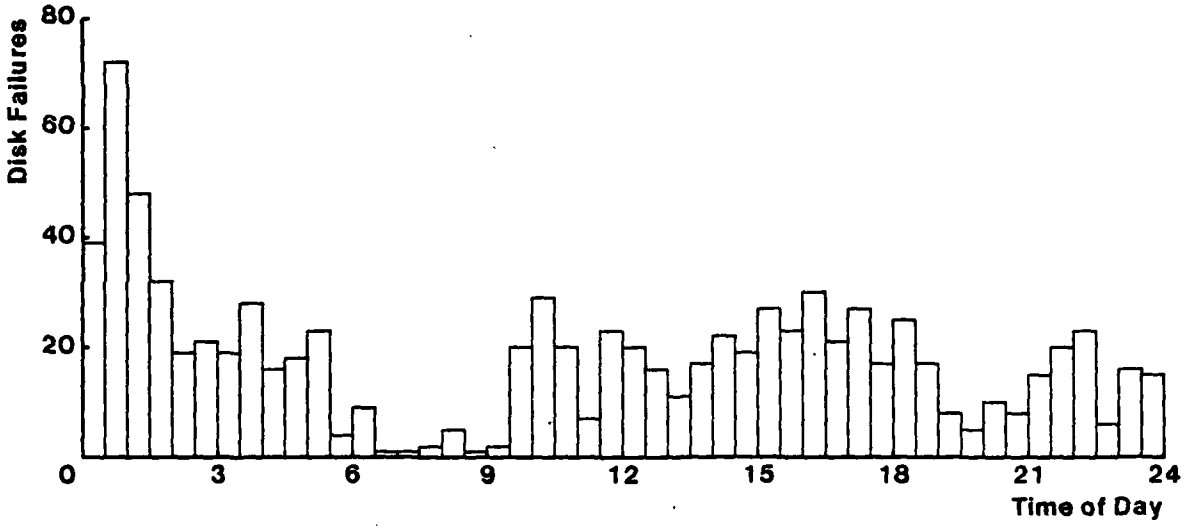


Figure 5-12: Histogram of disk failures as a function of time of day.

The hazard function is given by

$$h_{dk}(\tau) = \alpha_{dk} \cdot \sigma_{dk1}\left[1 - e^{-\beta_1 t}\right] \cdot \sigma_{dk2}\left[1 - e^{-\beta_2 t}\right] \cdot \frac{1}{\phi_{dk}(\tau)} \frac{\partial \phi_{dk}(\tau)}{\partial \tau} \quad (5.106)$$

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Cyclostat.	$s_c = 14.00$ $c_c = 2.01$	8	8.69	15.07	0.36
Stationary	$\alpha_c = 2.13$ $\sigma_{1c} = 1.42$ $\sigma_{2c} = 4.03$ $\beta_1 = 0.59$ $\beta_2 = 0.21$	6	8.642	12.592	0.19

Table 5-2: Results of applying a χ^2 goodness-of-fit test for the Cyclostationary and Stationary models with the file system failure data. The hypothesis that the models are good abstractions for the system behavior is confirmed since the level of confidence is larger that 0.05 in both cases.

Table 5-2 gives the results of applying a χ^2 goodness-of-fit test to the file system failure data. Again, although the Cyclostationary model gives a superior level of confidence the Stationary approximation also preforms very well. Therefore, if great accuracy is not necessary, some of the complexity involved in the manipulation of the cyclostationary expressions can be saved by neglecting the periodic component. Figure 5-13 shows the hazard functions of for both the Cyclostationary and Stationary approximations. Note the small range of variability due to the periodic component of the failure rate.

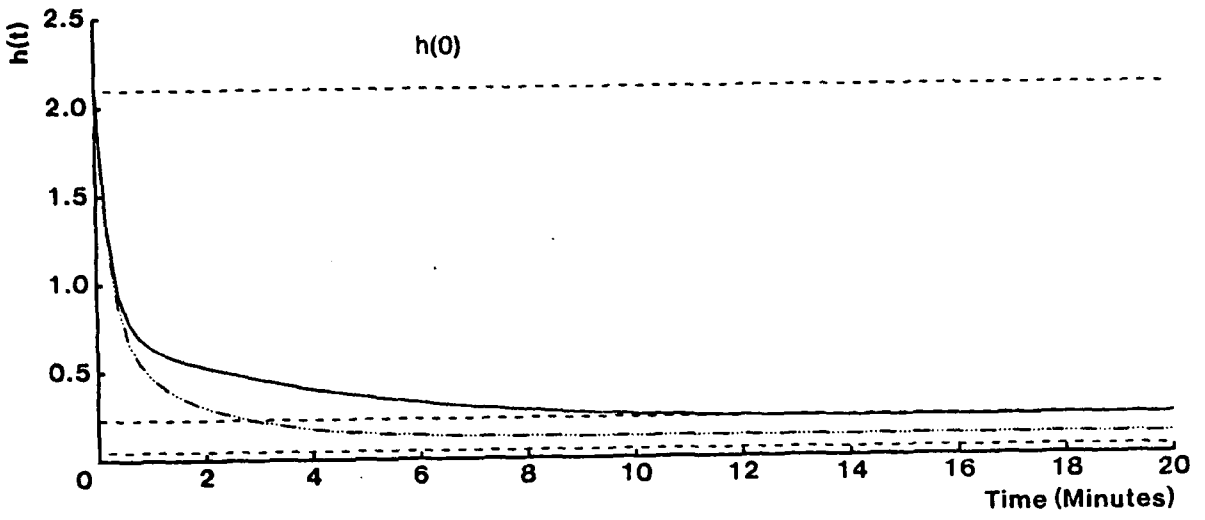


Figure 5-13: Hazard function of the equivalent non homogeneous Poisson process characterizing the statistics of the time to failure of a file system. Both hazard functions (according to the Cyclostationary and Stationary approximations) have been plotted. The Mean Time To Failure is 7 minutes, that would correspond to a constant hazard function of 0.7 according to the Exponential model. The two dashed lines at the bottom of the graph enclose the range of variability of the hazard function due to the periodic component of the failure rate $m^{dk}(t)$. Note that this range of variation can be neglected and that the main factor characterizing the hazard function is its decreasing effect due to the integral of the autocorrelation function $R_{x^{dk}x^{dk}}(\tau)$.

5.5. Summary

Both the Cyclostationary and Stationary models have been validated as suitable descriptions of failure processes in Time Sharing computers. Validation has been performed by applying χ^2 goodness-of fit tests to the PDF of the time to failure of each model with failure data obtained from a real system. Two failure processes have been used for this validation : a file system failure process, and the complete system failure process describing the statistics of the time to crash. The main conclusions are :

- Predominance of the decreasing hazard function effect due to the integrated autocorrelation function of the stochastic part of the failure rate.
- Marginal importance of the periodic component of the failure rate with respect to reliability prediction.

- Exponentially decreasing hazard function since the measured autocorrelation functions are exponentials.
- Predominance of the periodic component of the failure rate in the PDF of the system failure time as a function of time of day.

Obviously, if the decreasing rate of the hazard function is accepted to be exponential and the periodic component is neglected, it is not necessary to estimate the resource utilization functions. Instead, the values of α, σ_1, σ_2, β_1, and β_2 can be estimated directly from a history of failures. This is what was done with the Stationary approximations presented in this Chapter.

The properties of the Cyclostationary and Stationary models are further discussed in the following Chapter, where these two models are compared (numerically and qualitatively) with the other three models described in Chapter 2 : Exponential, Weibull, and Periodic.

Chapter 6
Discussion

6.1. Reliability modeling

The different models currently used to characterize the reliability of digital computing systems were summarized in Chapter 2. In this section, the predictions of those models, the predictions of the Cyclostationary and Stationary models, and the observed behavior of the system described in Chapter 4 are compared. In Section 6.1.1 the predictions of the different models with the observed system behavior are compared by means of numerical statistical tests. In Section 6.1.2 the assumptions made by each model are compared, along with some of their most general properties. The main conclusions of this Chapter are summarized in Section 6.3. The Reliability function and hazard function of each of the five models (Exponential, Weibull, Periodic, Cyclostationary, and Stationary) are summarized in Table 6-1.

6.1.1. Numerical comparisons : statistical tests

Table 6-2 shows the results of applying a χ^2 goodness-of-fit test between the actual failure data of the CMU-10A file system and the distributions predicted by the Exponential, Weibull, Periodic, Cyclostationary, and Stationary models using appropriate maximum likelihood estimates for each model. A χ^2 value smaller that 0.05 (i.e., a level of confidence greater than 0.05) indicates a good fit between predicted and observed behavior and suggests the acceptance of the hypothetical distribution as the real distribution characterizing the failure process.

As can be seen from Table 6-2 only the Cyclostationary and Stationary models show a clear good fit with the experimental data. Neither the Exponential nor the Periodic models seem to be able to describe the failure process with significant accuracy. The Weibull and simplified Stationary models (obtained by approximating the autocorrelation function by a single exponential) give levels of confidence close to 0.05, which suggests that these two models can be used when it is desired to trade some accuracy for model simplicity.

Exponential

$$R_e(\tau) = e^{-\lambda_e \tau} \tag{6.1}$$

$$h_e(\tau) = \lambda_e \tag{6.2}$$

Weibull

$$R_w(\tau) = e^{-(\lambda_w \tau \alpha_w)} \tag{6.3}$$

$$h_w(\tau) = \frac{\alpha_w \lambda_w}{(\lambda_w t)^{1-\alpha_w}} \tag{6.4}$$

Periodic

$$R_p(\tau) = e^{-\lambda_p \tau} e^{-F_p u(\tau)} \tag{6.5}$$

$$h_p(\tau) = \left[\lambda_p + F_p \frac{\partial u(\tau)}{\partial \tau} \right] \tag{6.6}$$

Cyclostationary

$$R_c(\tau) = e^{-(\lambda_c + \sigma_{c1} + \sigma_{c2})\tau - \frac{\sigma_{c1}}{\beta_1}[1 - e^{-\beta_1 \tau}] - \frac{\sigma_{c2}}{\beta_2}[1 - e^{-\beta_2 \tau}] + \ln \phi(t)} \tag{6.7}$$

$$h_c(\tau) = \lambda_c - \sigma_{c1}[1-e^{-\beta_1 \tau}] - \sigma_{c2}[1-e^{-\beta_2 \tau}] - \frac{1}{\phi(t)} \frac{\partial \phi(t)}{\partial t} \tag{6.8}$$

Stationary

$$R_s(\tau) = e^{-(\alpha_s - \sigma_{c1} - \sigma_{c2})\tau - \frac{\sigma_{s1}}{\beta_1}[1 - e^{-\beta_1 \tau}] - \frac{\sigma_{s2}}{\beta_2}[1 - e^{-\beta_2 \tau}]} \tag{6.9}$$

$$h_s(\tau) = \alpha_s - \sigma_{s1}[1 - e^{-\beta_1 \tau}] - \sigma_{s2}[1 - e^{-\beta_2 \tau}] \tag{6.10}$$

Table 6-1: Reliability and Hazard functions of the five compared models.

The Cyclostationary model, taking into account both the periodic workload component and the integrated autocorrelation function, gives the best description of the failure process. Figure 6-1 shows the hazard functions of the above five models in the case of file system failures.

Table 6-3 gives the results of applying a χ^2 test to the five models in the case of system failures (crashes). Again, only the Cyclostationary and Stationary models give levels of confidence larger than 0.05. The hazard functions of the five models are shown in Figure 6-2.

REMARK: Note that the predominant effect is that of having a decreasing hazard function due to the integrated autocorrelation function. Indeed, neglecting the periodic component still leads to an acceptable level of confidence for the Stationary model. On the other hand, neglecting the integrated autocorrelation function and taking into account only the periodic workload component leads to a characterization that has to be rejected, as the level of confidence of the Periodic model indicates.

6.1.2. Qualitative comparisons

As it has been shown in the previous section, the methodology presented in this thesis seems to lead to a more accurate characterization of system reliability than other more traditional models. Its *widespread use, however, is doubtful due to the complexity of the math involved.* Although the relevance of the results presented in this thesis is discussed in Section 5.3. and later on in Chapter 7, a comparison of the implicit assumptions and general properties of each model may help to decide when each model is appropriate.

6.1.2.1. Failure rate

Table 6-4 lists the assumptions made by each model concerning the failure rate of digital computing systems. The main difference between the Cyclostationary and Stationary models and the three traditional models is that traditional models assume the failure rate to be a deterministic function of time, while the Cyclostationary and Stationary models assume the failure rate to be a stochastic process.

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Exponential	$\lambda_e = 0.67$	7	130	14.067	0
Weibull	$\lambda_w = 0.91$ $\alpha_w = 0.68$	8	17.717	15.507	0.026
Periodic	$s_p = 1.25$ $c_p = 0.28$	12	1007	21.026	0
Cyclostat.	$s_c = 14.00$ $c_c = 2.01$	8	8.69	15.07	0.36
Stationary	$\alpha_s = 2.13$ $\sigma_{s1} = 1.42$ $\sigma_{s2} = 4.03$ $\beta_1 = 0.59$ $\beta_2 = 0.21$	6	8.642	12.592	0.19
Stationary (Simplified)	$\alpha_s = 1.69$ $\sigma_{s1} = 1.38$ $\beta_1 = 1.38$	8	19.434	15.507	0.013

Table 6-2: Results of a χ^2 goodness-of-fit test with the Exponential, Weibull, Periodic, Cyclostationary, and Stationary models for file system failures. Only the Cyclostationary and Stationary models give levels of confidence greater than 0.05. The Weibull and simplified Stationary models give smaller levels of confidence but close to 0.05. The hypothesis that the time to failure can be characterized with Exponential or Periodic models has to be rejected. The data used was obtained from five weekdays of system operation during which 877 (transient) failures were detected. The MTTF value is 7 minutes. The file system is composed of 8 RP06 disk drives totaling 1600 megabytes of on line storage.

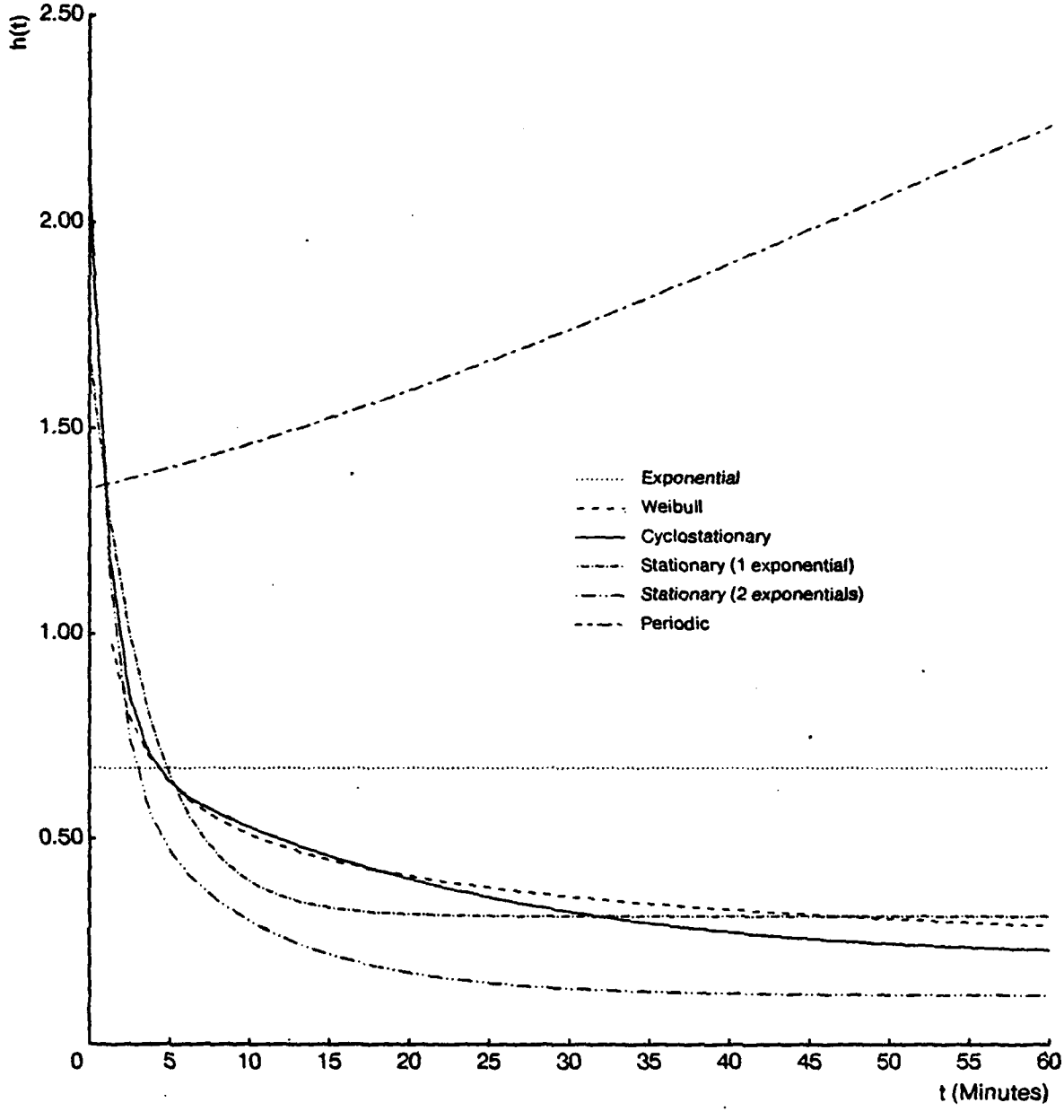


Figure 6-1: Hazard functions predicted by Exponential, Weibull, Periodic, and Cyclostationary models for file system failures.

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Exponential	$\lambda_e = 0.0097$	18	120	28.869	0
Weibull	$\lambda_w = 0.0137$ $\alpha_w = 0.61$	17	28	27.587	0.045
Periodic	$s_p = 0.01172$ $c_p = 0.0074$	17	119	27.587	0
Cyclostat.	$s_{sy} = 2.23$ $c_{hw} = 0.0082$	18	15.89	28.869	0.6
Stationary	$\alpha_s = 0.082$ $\sigma_{s1} = 0.073$ $\sigma_{s2} = 0.00413$ $\beta_1 = 0.285$ $\beta_2 = 0.0039$	14	15.89	23.685	0.3
Stationary (1 exp.)	$\alpha_s = 0.074$ $\sigma_{s1} = 0.067$ $\beta_1 = 0.22$	17	13.61	27.587	0.7

Table 6-3: Results of a χ^2 goodness-of-fit test with the Exponential, Weibull, Periodic, Cyclostationary, and Stationary models for system failures (crashes). Again, the cyclostationary and Stationary models give the best fit. The data used was obtained from 6 months of system operation during which 243 crashes due to transients or software were detected (Nov. 1979 to Apr 1980). The MTTS (Mean Time To reStart) value is 9 hours.

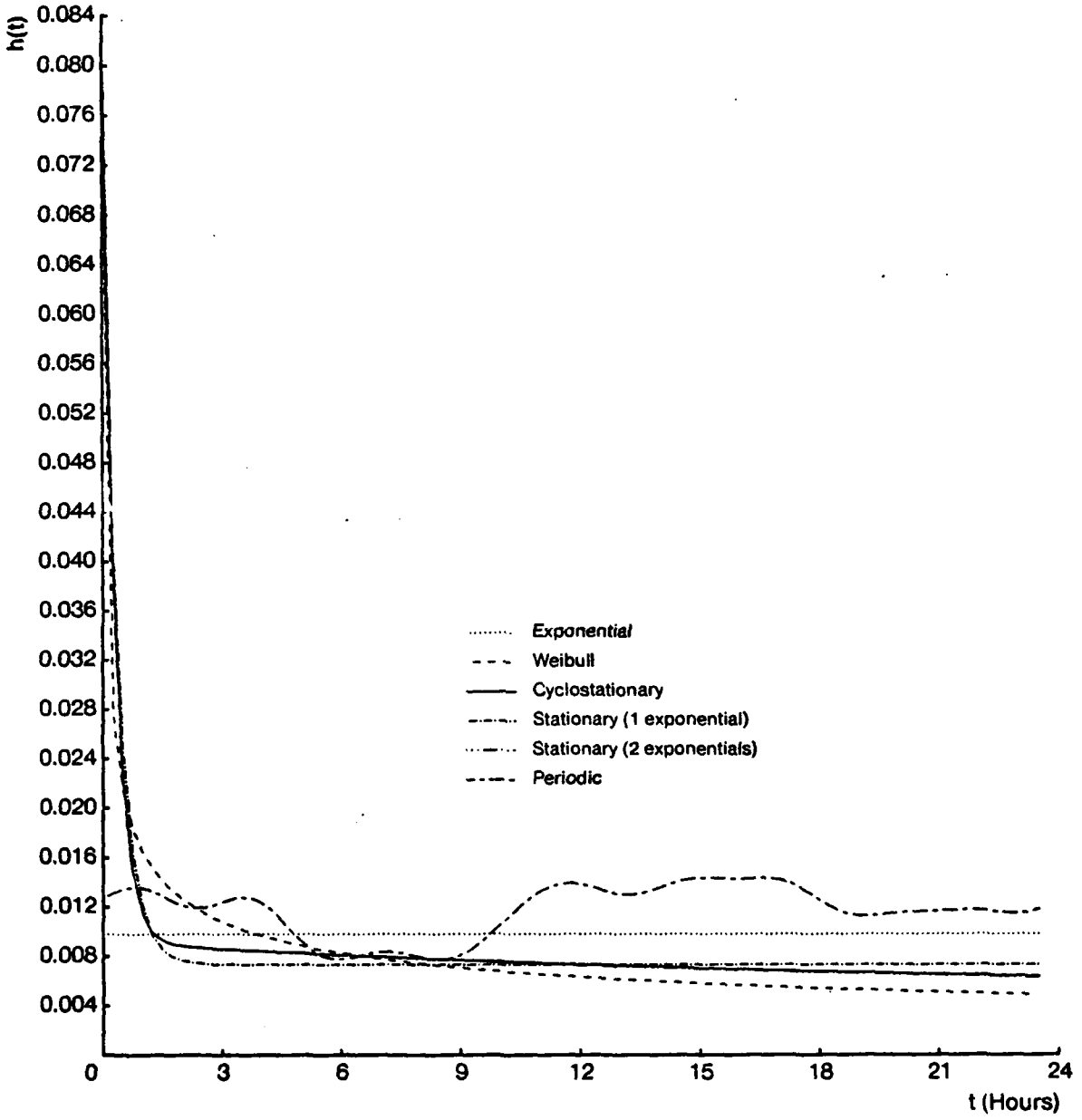


Figure 6-2: Hazard functions predicted by Exponential, Weibull, Periodic, and Cyclostationary models for system failures.

Model	Failure Rate		Hazard Function	
Exponential	λ	Constant	λ	Constant
Weibull	$\frac{\lambda\alpha}{(\lambda t)^{\alpha}}$	Decreasing	$\frac{\lambda\alpha}{(\lambda t)^{\alpha}}$	Decreasing
Periodic	m(t)	Periodic	m(t)	Periodic
Cyclostationary	$m(t) + x_t$	Cyclost. process	$k\, m(t)\, \eta(t)$	Decr. modulated by per. function
Stationary	$m + x_t$	Stat. process	$\eta(t)$	Decreasing

Table 6-4: Failure rates and hazard functions assumed by each of the five models

6.1.2.2. Hazard function

Loosely speaking, the difference between failure rate and hazard function is the difference between what actually *happens* and what can be easily *observed*. The evaluation of the exact failure rate at a particular time in a computer may be an interesting mathematical exercise (that of statistical inference of the value of a random variable from some of its after effects, i.e., failures). But conceptually, reliability characterization is easier in terms of the hazard function.

This distinction between failure rate and hazard function is not usually made in the Exponential, and Weibull models. Failure rate and hazard function are identified with the same time functions for those two models. A hazard function can be derived for the periodic model by averaging the value of the failure rate for all possible system starting times (a simple calculation will show that the hazard function for the periodic model is proportional to the squared failure rate).

Recall that if h(t) is the hazard function, $h(t)\Delta t$ is the probability of observing a failure in the infinitesimal interval $[t, t+\Delta t]$. Thus, for the exponential model any interval has the same probability of

containing a failure. For the Weibull model, the probability decreases with time. For the periodic model this probability is also periodic. Both for the cyclostationary and stationary models this probability is decreasing. The point is that for the cyclostationary and stationary models the hazard function has been obtained after computing the expectation for *all possible realizations of the failure rate*.

Therefore, it is not maintained here that the probability of observing a failure in an infinitesimal interval actually decreases with time. What is maintained here is that if the behavior of many systems is observed, or if the behavior of a single system is observed for a sufficiently long time interval, the measured parameters will look *as if* the infinitesimal probability would decrease with time. But the actual infinitesimal probability for a particular system at a particular moment in time is a random variable, namely, its failure rate at that moment.

6.1.2.3. Reliability Function

Further insight into the implicit implications of using each of the five models can be gained by comparing their Reliability functions. Recall from Chapter 2 that the Reliability function is the probability that no failure will be observed before time t. Only three Reliability functions will be compared : Exponential, Weibull, and Stationary, given in (6.1), (6.3), and (6.9). Figure 6-3 shows the above three reliability functions for the file system failure data. Only these three models are compared to provide a clear idea of their main differences and similarities. The Exponential model is the most widely used in reliability theory. The Stationary model gives a good fit with experimental data while not being as complex as the Cyclostationary model. And the Weibull model is the closest previous approximation to the methods presented in this thesis. Note from Figure 6-3 that for values of t smaller than 14 minutes (about twice the MTTF value) the Stationary and Weibull models essentially agree in their predictions while the Exponential model predicts reliability values larger than the other two models. For values of t larger than 14 minutes, the Exponential model predicts reliability values smaller than the predictions of the Stationary and Weibull models, the larger predictions corresponding to the Weibull. Figure 6-4 shows the same three reliability functions for the case of system failures. Again, the Exponential model gives reliability predictions up to 20% larger than the other two models for small values of t, and too small reliability values for large values of t. In this case crossover occurs at t = 13 hours, about 1.5 times the MTTF value.

If the Stationary model is accepted as the best descriptor of the file system reliability (which is a reasonable thing to do after examining the values of the χ^2 test shown in Section 6.1.1) the following two conclusions are reached :

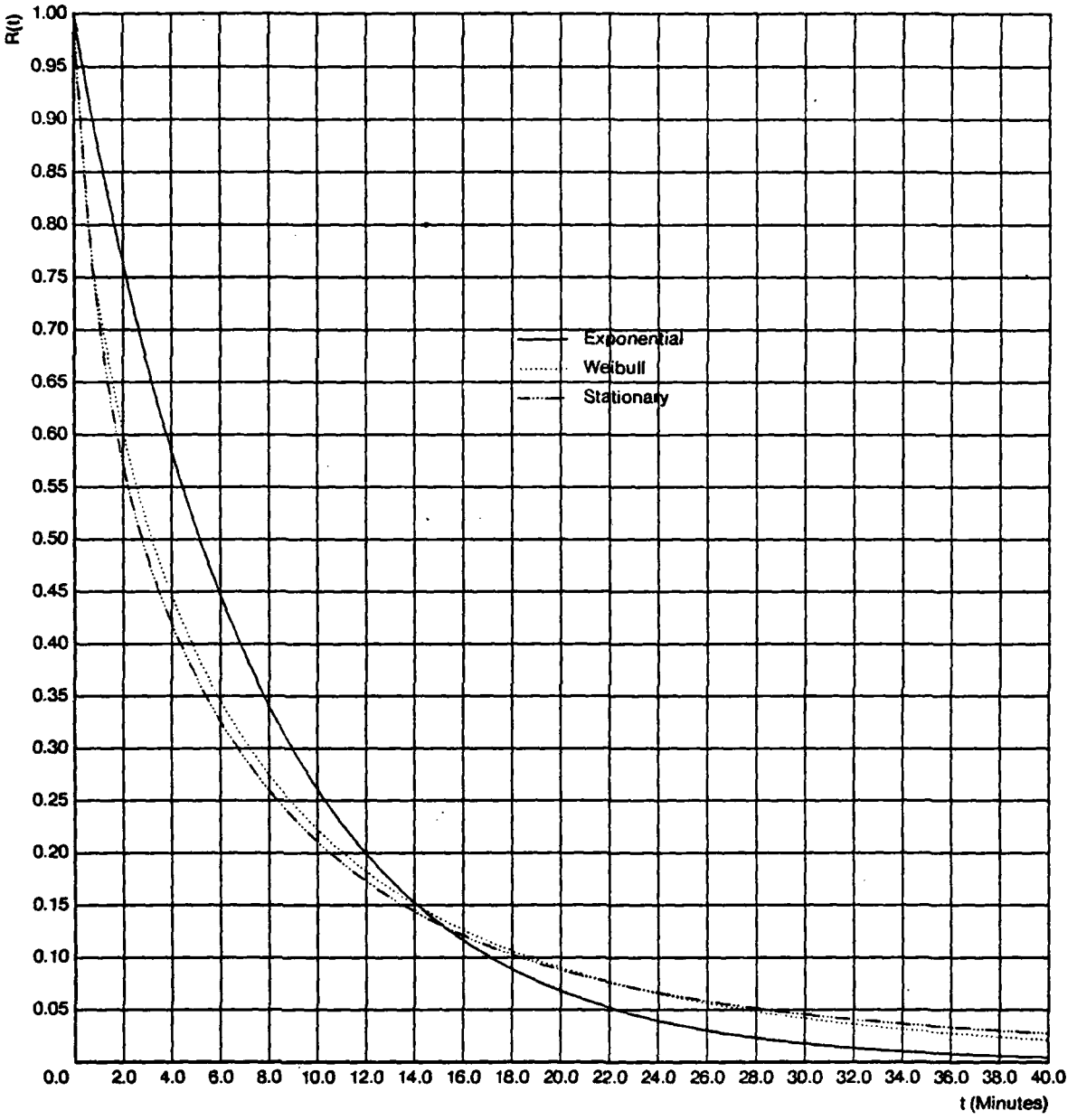


Figure 6-3: Reliability functions according to the Exponential, Weibull and Stationary models for file system transient failures.

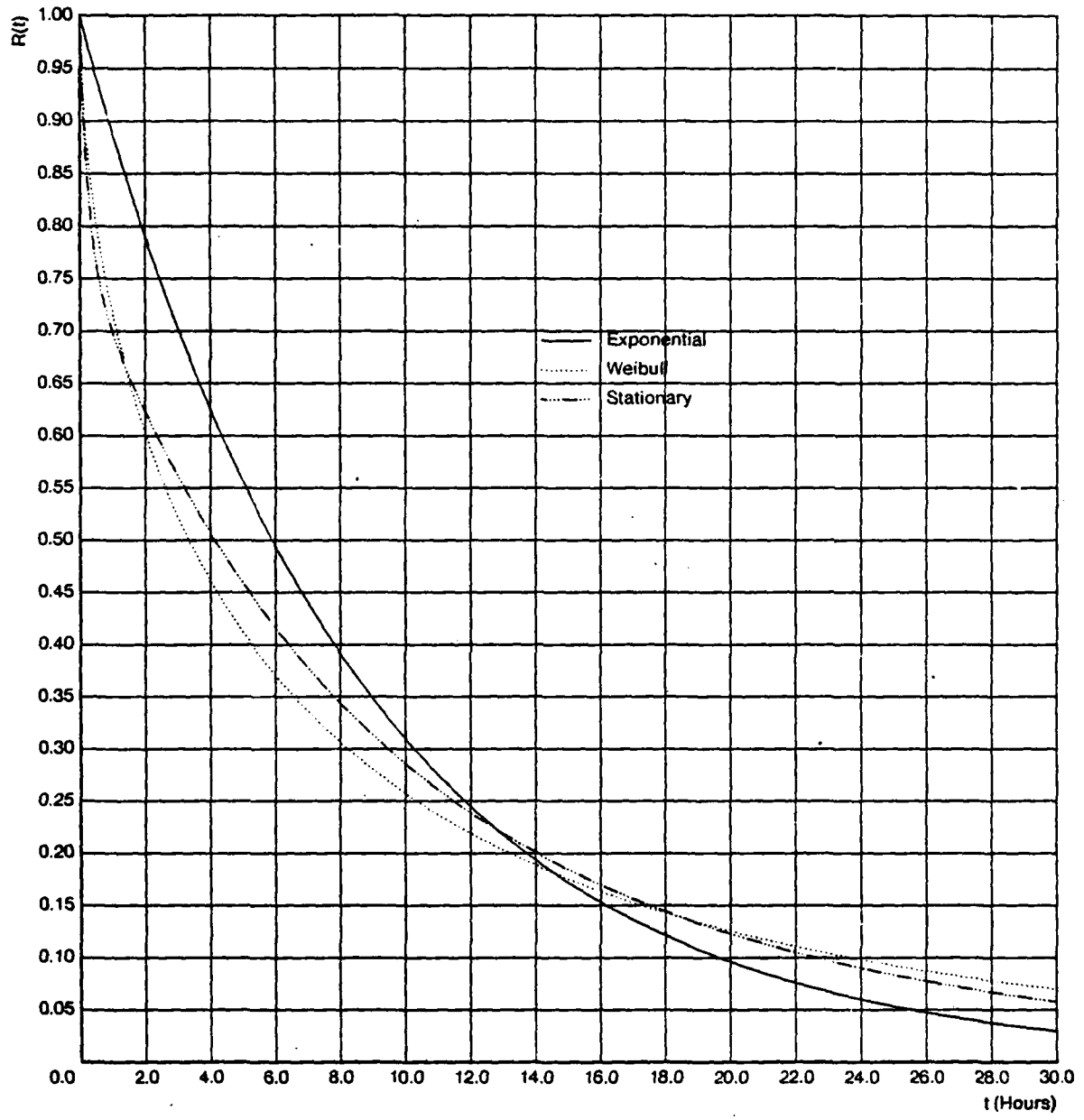


Figure 6-4: Reliability functions predicted by the Exponential, Weibull, and Stationary models for system failures (crashes).

- For small values of t, the reliability predictions of the Exponential model are essentially too optimistic. The actual reliability is lower than predicted by the Exponential model.
- For large values of t, the reliability predictions of the Exponential model too pessimistic. The system is actually more reliable than predicted by the Exponential model.
- The differences in reliability prediction are specially important for values of t smaller than the Mean Time To Failure, where the Exponential model differs by almost 20% with the Weibull and Stationary models.
- Reliability predictions of the Stationary and Weibull models are within 5% through all range of values of time.

Overall, the results presented here are consistent with the results presented in [McConnel 81]. This is important because the analysis done by [McConnel 81] with the Weibull distribution was extended to redundant systems (duplex, triplex and TMR) for which the same behavior was observed.

6.2. A possible new design parameter

Assume now that the autocorrelation function of the fraction of time in kernel mode is somehow under the control of system designers. Maximum reliability would be obtained if

$$R_{xx}(\tau)| = (\sigma_{sy_1} + \sigma_{sy_2}) \tag{6.11}$$

That is, the stochastic component of the system failure rate would be white noise. In this case, the PDF of the time to failure would become an exponential with parameter

$$\lambda = \alpha \cdot (\sigma_{sy_1} \cdot \sigma_{sy_2}) \tag{6.12}$$

The system would still be able to do the same amount of work in the sense that the average fraction of time in kernel mode is independent of the shape of its autocorrelation function.

If such an autocorrelation function could be obtained on the CMU-10A, the MTTF value would be 16 hours, compared with the real MTTF value of 9 hours obtained with an exponentially decreasing hazard function. Figure 6-5 shows the reliability functions obtained from the Stationary approximation considering both an exponentially decreasing hazard function and a delta function (white noise). Although pure white noise is impossible to obtain physically (it has an infinite bandwidth), the fact that faster decreasing rates for the autocorrelation function means also more reliable systems is a new factor to take into account.

Recall from Section 4.1.2.1 that the distinctive property of white noise is that it is unpredictable. Thus, designing a system with a faster decreasing hazard function means also removing some predictability from its behavior. Clearly, this indicates a tradeoff between performance and reliability since many algorithms to enhance system performance (for instance, in schedulling and paging) are precissely based on predicting future system behavior. However, it is not clear at present how the rate at which the hazard function decreases (that is, the shape of the failure rate autocorrelation function) can be controlled.

6.3. Summary

Three different known model used to characterize the reliability of digital computers have been compared with the two main modeling methods presented in this Thesis (Cyclostationary and Stationary) and with actual failure data collected on the computing system described in Chaper 5. Statistical test performed with two different failure processes clearly suggest :

- Acceptance of the Cyclostationary and Stationary modeling methods as suitable tools to characterize system reliability.
- Rejection of the Exponential and Periodic models as accurate descriptions of computers failure processes. The only exception may be the use of the Exponential model when simplicity has absolute priority.
- Acceptance of the Weibull and simplified Stationary models as marginally accurate descriptions of system reliability. They are not as good as the Cyclostationary or stationary models nor as bad as the Exponential or Periodic.
- Introduction of a possible new design parameter: the rate at which the hazard function decreases to its asymptotic value.

Qualitative comparisons between the Exponential model and the Stationary and Weibull models have confirmed the findings of [McConnel 81], that is,

- The Exponential model is too optimistic when predicting reliability for small values of t.
- The Exponential model is too pessimistic when predicting reliability for large values of t. The Weibull model has been found too optimistic when predicting reliability for large t.

Clearly, the validity of a modeling methodology cannot be confirmed or denied by the results of a single experiment. However, the results obtained so far are encouraging and justify a more detailed study. Therefore, the next Chapter is dedicated to elaborate some applications derived from the

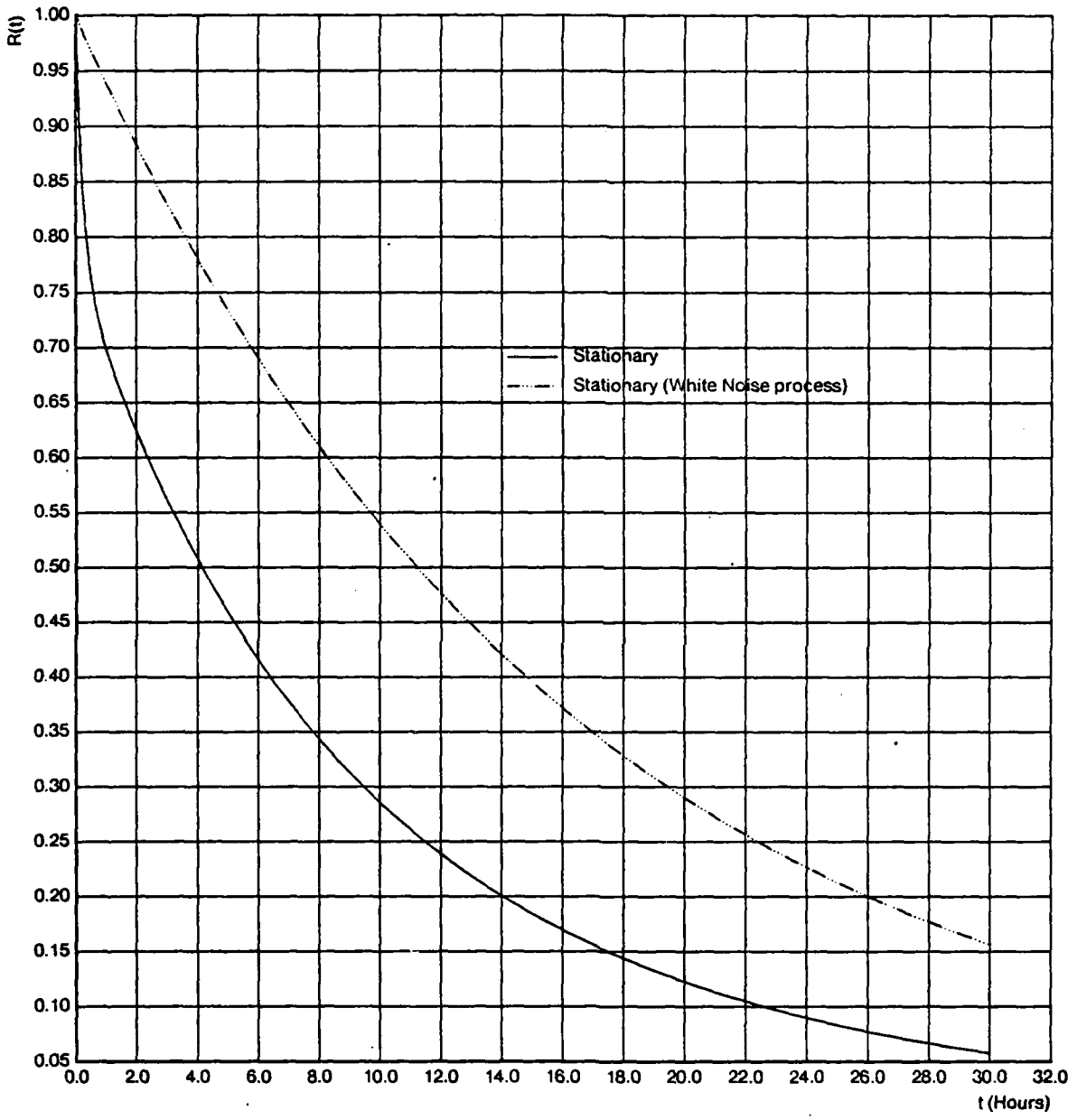


Figure 6-5: Reliability functions obtained from the Stationary model by considering the real autocorrelation function and white noise.

Cyclostationary and Stationary modeling methods. In some cases, these applications will be independent of the results obtained until now. However, they are included because they are natural extensions to the philosophy used through the Thesis.

Chapter 7
Applications

The previous Chapters have shown how the Cyclostationary and Stationary modeling methods are the correct approach to characterize computing systems reliability operating under periodic or constant workload respectively. Further, it has been shown how the Stationary model may be just enough to predict reliability even for systems under periodic workload, since the effect of having a periodic workload in the failure rate is minor compared with the effect of considering the failure rate to be a Gaussian process, and therefore having a decreasing hazard function.

Nevertheless, as will be shown in Sections 7.1 through 7.3, that workload periodicity can still be used to obtain some new results related to reliability characterization. The first contribution is presented in Section 7.1, where it is shown how the contributions of software and hardware errors can be easily evaluated.

It was stated in Chapter 2 that one of the main problems associated with the acceptance of fault-tolerance as a more desirable attribute of general purpose computing systems was the fact that performance evaluation and reliability characterization are unconnected. Thus, in Section 7.2 an attempt is made to elaborate an integrated Performance/Reliability model.

In Section 7.3 the problem of determining the optimum checkpointing interval in a transaction processing system is revisited and refined. The purpose of this section is to determine if the modeling methods presented in this thesis in any way invalidate or confirm previously obtained results.

Finally, in Section 7.4 a first step is given in a completely new area: modeling the effects of hardware transients, software faults, *and permanent hardware faults*. The main conclusions of the Chapter are summarized in Section 7.5.

7.1. The impact of unreliable software on the observed system reliability

As it was described in Chapter 2, current software reliability modeling and measurement efforts concentrate in the evaluation of static software attributes. Some of these attributes are the number of bugs present in a software package, or the mean time between software failures of a set of programs operating in a controlled environment. Here, however, the evaluation will refer to the observed behavior of systems operating in the field under dynamically changing conditions. Further, software reliability models usually refer to parameters of interest to the software development team, while here an effort will be made to quantify the impact of software unreliability to the average user of a Time Sharing system and to the user community.

Perhaps the simplest question to be asked is whether a given system failure is due to a hardware transient or to a software fault. Most operating systems provide some tools to help answering such question. The most primitive tool is just a memory dump that has to be manually analyzed to resolve the cause of the failure. Other systems provide more information in an error log. And some systems even attempt to automatically classify all failures. However, some experience using such tools soon teaches the difficulty of the problem. Except for a few clear hardware failures (a hard memory parity error while accessing one of the kernel data structures) most failures usually remain unresolved. Assume that a system is hung in an infinite loop in the kernel and the system has to be manually crashed by the operator. How can it be known if a part of code was overwritten by the software itself or if an undetected transient altered the destination address of a jump instruction?

The method proposed here to resolve such ambiguities is probabilistic. Although each system failure is due to a particular cause, to learn the exact cause for each failure with a reasonable level of confidence may be extremely costly. The method proposed here will give only expectations and averages. But it is substantially cheaper.

It was shown in Section 5.3.3 how the instantaneous system failure rate at each moment in time is given by

$$\lambda_t^{sy} = c_{hw} + [s_{hw} + s_{sw}(t)]k_t \qquad (7.1)$$

which can be viewed as the superposition of the hardware and software failure rates.

$$\lambda_t^{sw} = (s_{sw_1} m(t) + s_{sw_2}) k_t \tag{7.2}$$

$$\lambda_t^{hw} = c_{hw} + s_{hw} k_t \tag{7.3}$$

Since $k_t = m(t) + x_t$, it is convenient at this point to indicate the dependency on x_t more explicitly

$$\lambda_t^{sw}(x_t) = (s_{sw_1} m(t) + s_{sw_2})(m(t) + x_t) \tag{7.4}$$

$$\lambda_t^{hw}(x_t) = c_{hw} + s_{hw}(m(t) + x_t) \tag{7.5}$$

such that the *system* failure rate is given by

$$\lambda_t^{sy}(x_t) = \lambda_t^{sw}(x_t) + \lambda_t^{hw}(x_t) \tag{7.6}$$

The system failure process can therefore be viewed as a *marked* doubly stochastic Poisson process, each failure being associated with a *mark* specifying if it is hardware or software related. Given that a failure has occurred at time t_f, the probability that this failure is due to software is

$$p_{sw}(t_f) = E\left\{ \lambda_{t_f}^{sw}(x_{t_f}) \frac{1}{\lambda_{t_f}^{sw}(x_{t_f}) + \lambda_{t_f}^{hw}(x_{t_f})} \right\} \tag{7.7}$$

where the expectation is taken with respect the statistics of x_{t_f} and

$$p_{hw}(t_f) = 1 - p_{sw}(t_f) \tag{7.8}$$

Hence,

$$p_{sw}(t_f) = \frac{1}{(2\pi)^{1/2}\sigma_x} \int_{k_{min}-m(t_f)}^{\infty} \frac{(s_{sw_1} m(t_f) + s_{sw_2})[m(t_f) + u]}{c_{hw} + [s_{hw} + s_{sw_1} m(t_f) + s_{sw_2}][m(t_f) + u]} e^{-u^2/2\sigma_x^2} du \tag{7.9}$$

where the restriction of having a strictly positive failure rate has been taken care of in the lower limit of the integral.

Figure 7-1 shows the probability that a crash is due to a software error as a function of the time of day for the CMU-10A after computing the maximum likelihood values of the coefficients according to Section 5.3.4. Since the linear term in the failure rate s_{sy} cannot be separated in its software and hardware components (s_{sw_1} and s_{hw}) Figure 7-1 shows the upper and lower bounds obtained by assuming $s_{sw_1} = s_{sy}$ and $s_{sw_1} = 0$. On the average, it seems that software accounts for 60% of the crashes while the remaining 40% is due to hardware. This is a misleading interpretation because the

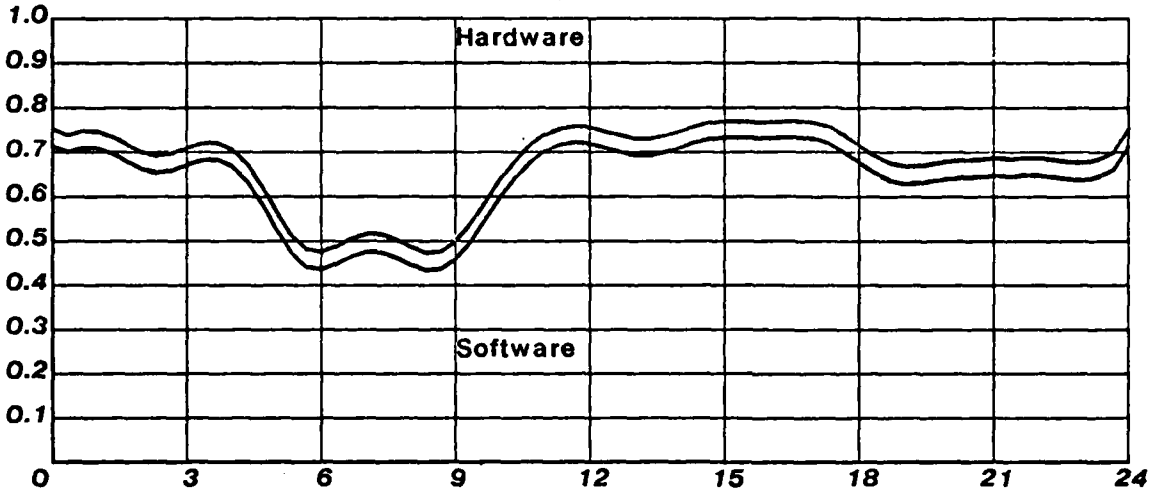


Figure 7-1: Probability that a crash is due to software or hardware as a function of the time of day

system does not crash at any time with equal probability. Consider a set of failures observed at times $t_{f_1},\ldots,t_{f_N}$. The expected number of failures due to software is

$$E[n_{sw}|N;t_{f_1},\ldots,t_{f_N}) = \sum_{k=1}^{N} p_{sw}(t_{f_k}) \tag{7.10}$$

This expectation has been computed for the set of 243 crashes observed in six months of operation of the CMU-10A. Since the system crashes more often at the times that the contribution of unreliable software is larger, 67% of the crashes are due to software. But it is still possible to refine this number. The impact of each crash depends on the number of jobs being executed at the time of crash. Figure 7-2 shows the average number of jobs executing in the CMU-10A as a function of time of day. Given that a crash occurs at time t_f, the expected number of jobs crashed due to software, $E[J^{sw}]$, is

$$E[J^{sw}|t_f] = p_{sw}(t_f)\, E[J_{t_f}] \tag{7.11}$$

where $E[J_{t_f}]$ is the expected number of jobs executing at time t_f. Given a set of N failures at times $t_{f_1},\ldots,t_{f_N}$, the expected number of jobs aborted due to software is

$$E[J^{sw}|N;t_{f_1},\ldots,t_{f_N}] = \sum_{k=1}^{N} p_{sw}(t_{f_k})\, E[J_{t_{f_k}}] \tag{7.12}$$

The value obtained for the CMU-10A is that 70% of the jobs aborted in system crashes do so because of software errors. A percentage substantially higher than the 60% originally computed for the probability that a single crash is due to software. These results are summarized in Table 7-1.

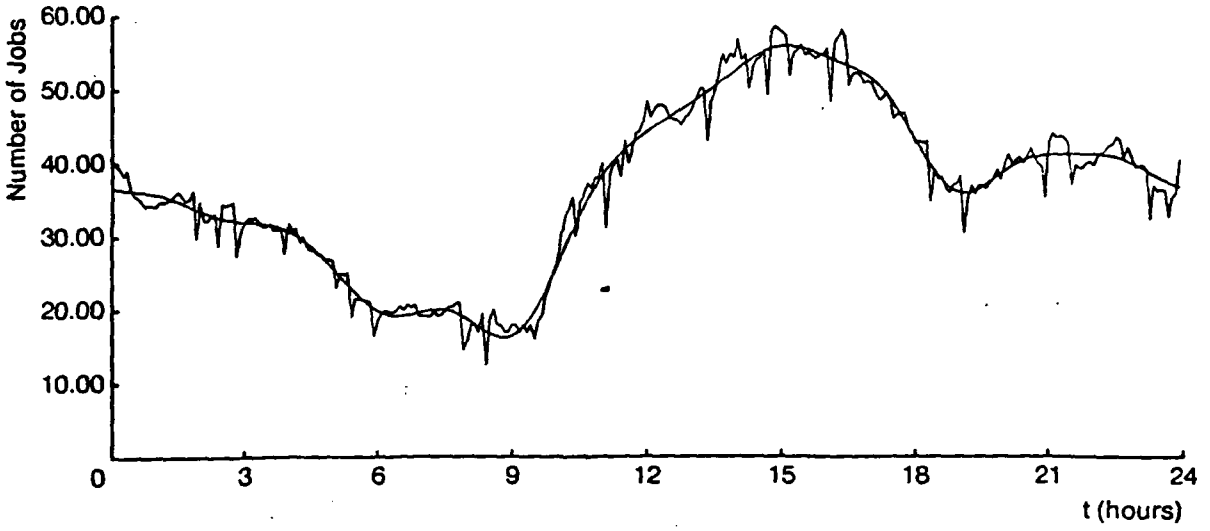


Figure 7-2: Average number of jobs executing as a function of time of day

Range of variability of the probability that a crash is due to software depending on the time of day at which the crash occurs	45%-75%
Probability that a crash is due to software averaged over one day period	60%
Expected percentage of crashes due to software during 6 months of operation of the CMU-10A	67%
Expected percentage of jobs aborted due to software during 6 months of operation of the CMU-10A	70%

Table 7-1: Different views of the impact of software in system unreliability

7.2. Performance/Reliability evaluation

7.2.1. The user's viewpoint

From a user's viewpoint, working with an unreliable system has an added cost that would not be present in a failure free system. This added cost due to unreliability is mainly due to two factors:

- A possible delay in finishing the user's task. The system may fail, remain unavailable for a while, and parts of the programs being executed may have to be repeated afterwards. The expected time required to complete a task is therefore longer in an unreliable system than in a failure free system
- The cost associated with repeated computations. That is, the cost associated with the use of resources that effectively may be useless, since the system may fail and some computations may have to be repeated.

These costs will be quantified for a CMU-10A user in this section. The approach is essentially the same that as in [Castillo 80a]. The problem of evaluating the added cost due to unreliability is visualized in Figure 7-3.

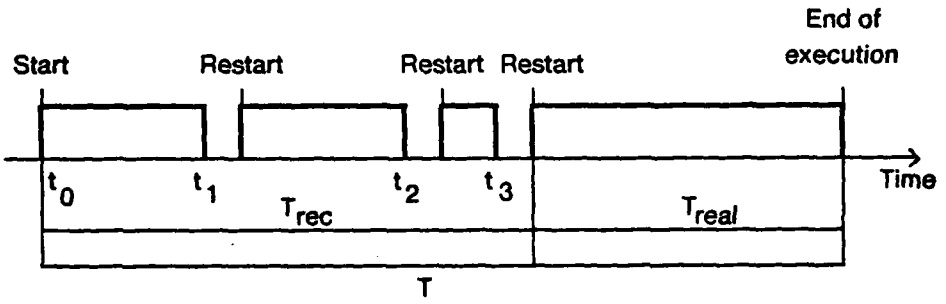


Figure 7-3: Typical system of events illustrating the unreliable behavior of a computing system from a user viewpoint

A program is started at time t_0 and failures occur at times t_1, t_2 ..., such that after each failure the

program has to be restarted. Complete execution terminates after the system has been operating continuously for a time T_{real}.

The total elapsed time since the user starts the program until the the program correctly completes execution, T, is equal to T_{real} plus T_{rec}, time during which the program was executing but wasted because the system failed before the program finished execution.

T_{real} is a random variable whose statistics depend on the resources needed by the program to complete execution (CPU time, storage requirements, etc.) and on the system workload during program execution (i.e., at which rate are these resources provided by the operating system depending on competing requests by other users). T_{rec} is another random variable whose statistics depend on T_{real} and and on the statistics of the time to failure. The total expected cost (in terms of time) incurred in executing the program is

$$E[C^T] = E[T_{rec}] + E[T_{real}] \tag{7.13}$$

where the first term in (7.13) is obviously the added cost due to unreliability in the sense that it would be zero if the program were executed in a failure free system. The failure process will be assumed to be stationary and the average workload will be assumed to be constant. The expected cost is then given by

$$E[C^T] = \int_0^\infty p_{T_{real}}(x)\, E[T_{rec}|T_{real} = x]\, dx + E[T_{real}] \tag{7.14}$$

Given T_{real}, the expected value of T_{rec} is

$$E[T_{rec}|T_{real} = x] = \sum_{n=1}^{\infty} P(N_f = n|T_{real} = x)\, E[T_{rec}|T_{real} = x; N_f = n] \tag{7.15}$$

$P(n_f = n|T_{real} = x)$ is the probability that the program is restarted n times given that it requires x units of time of continuous system operation and is given by given by

$$P(n_f = n|T_{rel} = x) = [P_f(\tau<x]^n P(\tau>x) \tag{7.16}$$

If t_r is the time from restart to failure,

$$E[T_{rec}|T_{real} = x; N_f = n] = n\, E[t_r|T_{real} = x] \tag{7.17}$$

Substituting now (7.17) and (7.16) in (7.15)

$$E[T_{rec}|T_{real} = x] = E[N_f|T_{real} = x]\, E[t_r|T_{real} = x] \tag{7.18}$$

That is, the expected value of T_{rec} is equal to the expected number of failures multiplied by the expected time from restart to failure given that $T_{real} = x$. The expected number of failures is

$$E[N_f|T_{real} = x] = \sum_{n=0}^{\infty} n\,[P_f(\tau<x)]^n\,P_f(\tau>x) \tag{7.19}$$

$$= \frac{P_f(\tau<x)}{P_f(\tau>x)} \tag{7.20}$$

The distribution of the time from restart to failure given that a failure occurs before x units of time is equal to the distribution of the time to failure truncated at time $\tau = x$, that is,

$$p_{t_r}(t|T_{real} = x) = \frac{1}{P_f(\tau<x)}\; p_{t_f}(t)\,[U(t) - U(t-x)] \tag{7.21}$$

where $p_{t_f}(t)$ is the pdf of the time to failure and U(t) is the step function

$$U(t) \begin{cases} 1 & \text{if } t>0 \\ 0 & \text{otherwise} \end{cases} \tag{7.22}$$

Therefore,

$$E[t_r|T_{real} = x] = \int_0^{\infty} t\,p_{t_r}|T_{real}(t|T_{real} = x)\,dx \tag{7.23}$$

$$= \frac{1}{P_f(\tau<x)}\left[\,E[t_f] - E[t'_f(x)]\,\right] \tag{7.24}$$

where

$$E[t'_f(x)] = \int_x^{\infty} t\,p_{t_f}(t)\,dt \tag{7.25}$$

Substituting now (7.24) and (7.20) in (7.18) the following result is obtained

$$E[T_{rec}] = E[t_f]\int_0^{\infty} \frac{p_{T_{real}}(x)}{P_f(\tau<x])}\,dx \cdot \int_0^{\infty} \frac{p_{T_{real}}(x)\,E[t'_f(x)]}{P_f(\tau<x)}\,dx \tag{7.26}$$

Figure 7-4 shows the expected elapsed time required to execute a program at three different times of day for different values of T_{min}. For each curve, the straight line represents the second term in equation (7.13), that is, it is the expected elapsed time due to workload only. The solid line represents the total expected elapsed time. At 12:00, the contribution of unreliability to the expected elapsed time of a program requiring 30 minutes of CPU is of 30%. The curves were obtained by actually measuring the distribution of the elapsed time required to execute a CPU bound program at the three times of day in the absence of errors. The mean time to failure at each time of day was measured by counting the number of crashes occurred in two hour time slots centered at each of the three times considered.

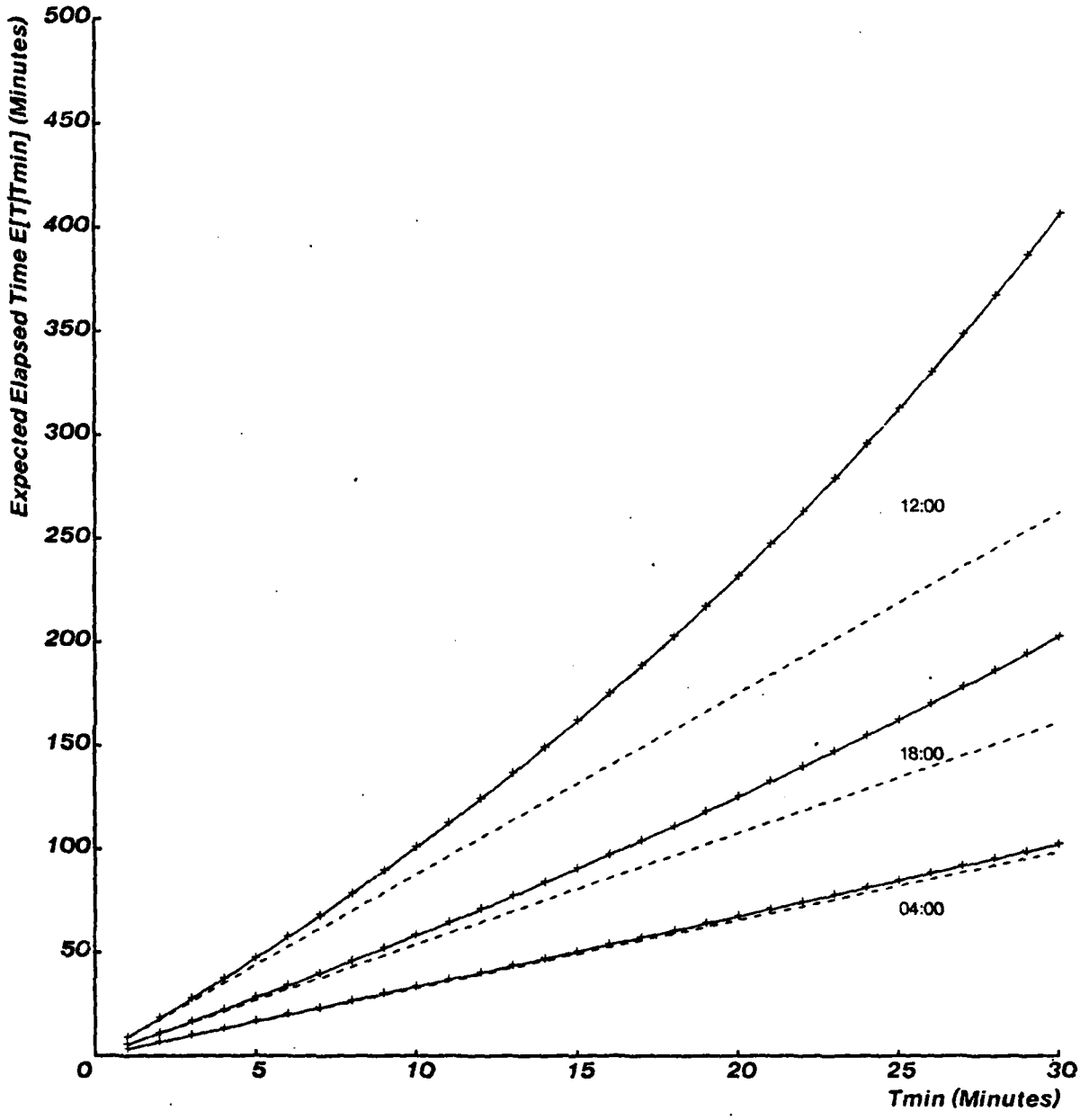


Figure 7-4: Expected elapsed time required to execute a program at three different times of day

7.2.2. The manager's viewpoint

In the previous section it has been shown how the added cost due to unreliability can be evaluated for a user trying to complete a task given the resources needed by his task and the system workload patterns. Here, a measure will be developed of potential use to system administrators. The idea is to evaluate the cost due to unreliability of a computer system operating as a server of computing utility.

Let J_t be the number of active jobs at time t. For fixed t, J_t will be, in general, an integer valued random variable. J_t is therefore an integer valued stochastic processes. Assume that a system failure occurs at time t_{f_1}. The added cost due to that failure can be evaluated as,

$$C(t_{f_1}) = J_{t_{f_1}} C_d + \sum_{i=1}^{J_{t_{f_1}}} C_{r_i} \tag{7.27}$$

where $J_{t_{f_1}} C_d$ is the cost associated with the time that the system is down (which will be assumed to be fixed for failures due to transients and software) and C_{r_i} is the cost associated with the recovery of the i-th job. Assume now that the system has been operating for N_d days, during which N_f failures have occurred at times $t_{f_1},\ldots,t_{N_f}$. The *expected* added cost due to unreliability during these N_d days is

$$C(N_d|N_f;t_{f_1},\ldots,t_{N_f}) = E\Big\{ \sum_{k=1}^{N_f} J_{t_{f_k}} C_d \Big\} + E\Big\{ \sum_{k=1}^{N_f} \sum_{i=1}^{J_{t_{f_k}}} C_{r_i} \Big\} \tag{7.28}$$

If the recovery cost for any job is independent of the number of jobs active at the time of failure, and the C_{r_i} are assumed to be identically distributed random variables,

$$C(N_d|N_f;t_{f_1},\ldots,t_{N_f}) = \sum_{k=1}^{N_f} E\{J_{t_{f_k}}\} C_d + \sum_{k=1}^{N_f} E\{J_{t_{f_k}}\} E\{C_{r_i}\} \tag{7.29}$$

According to the results presented in Chapter 3, given that N_f failures have occurred, each of the t_{f_i} has a distribution over a one day period equal to the periodic component of the failure rate, $f(m(t),\vec{\kappa})$. Thus

$$C(N_d|N_f) = N_f \Big[C_d + E\{C_{r_i}\} \Big] \int_0^T f(m(t),\vec{\kappa})\, E\{J_t\}\, dt \tag{7.30}$$

where it has been assumed that a stationary distribution exists for the C_{r_i}. Finally, since neither C_{r_i}, m(t), or J_t depend on N_f,

$$C(N_d) = E\{N_f\} \Big[C_d + E\{C_{r_i}\} \Big] \int_0^T f(m(t),\vec{\kappa})\, E\{J_t\}\, dt \tag{7.31}$$

For a system administrator, the interesting question is whether the policies regulating the use of the system can be modified such that the above cost is minimum, while simultaneously executing, on the average, the same number of jobs per unit time.

$E\{N_f\}$ is the expected number of failures in N_d days and can be reduced only by improving the hardware or the operating system. C_d is the cost associated with the system down time after a failure, which presumably will already be as small as possible. $E\{C_{r_i}\}$ is the cost associated with the abortion of user jobs, and depends on users patterns of use, programming style, and so on. The only term left for the administrator to play with is the variable cost associated with the system workload variations. Let C_u be equal to the added cost depending on workload variations

$$C_u = \int_0^T f(m(t),\vec{\kappa})\, E[J_t]\, dt \tag{7.32}$$

Since $f(m(t),\vec{\kappa})$ is a polynomial on $m(t)$, C_u will be minimum when $m(t) = \overline{m}$, the mean value of $m(t)$. Thus, allowing the workload to vary around its mean value has an associated cost in itself. For the CMU-10A the periodic variations actually increase the cost due to unreliability by 8% in the sense that $C_u = 0.13$ and for constant workload $C_u^{min} = 0.12$. Obviously, the number of jobs processed in one day is the same in both cases.

Dividing (7.31) by N_d, the added cost due to unreliability per unit time is obtained

$$C = \frac{1}{MTTF} \left[C_d + E\{C_{r_i}\} \right] C_u \tag{7.33}$$

This expression is important as it includes all factors for which unreliability has an associated added cost. From (7.33) is seen that doubling the MTTF value actually decreases the added cost in half. It has already been shown how C_u increases this cost as a consequence of workload periodicity. Finally, C_d is usually going to be small compared with $E\{C_{r_i}\}$, the recovery cost associated with each job. Thus, one way to reduce the added cost due to unreliability is to reduce the expected recovery cost from failures. The next section shows how the recovery costs can be reduced by introducing checkpointing.

7.3. On the optimum checkpointing interval

To diminish the added cost due to unreliability several alternatives are possible according to expression (7.31). Assuming that hardware and operating system reliability are given and that the workload patterns cannot be changed there is still a way by which the cost associated with delays and repeated computations can be reduced. Assume that at certain points in time called *checkpoints* a copy of the program memory image and data structures is made and stored in some secondary storage medium. Figure 7-5 shows a typical sequence of events when checkpointing is possible. If a failure occurs before the program completes execution, the copy of the program image at the most

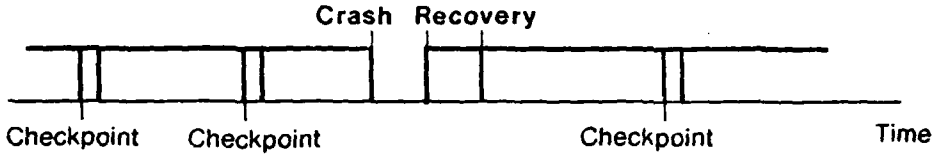


Figure 7-5: Typical sequence of events in a system with checkpointing facilities. The total added cost due to unreliability is the cost associated with the checkpoint operation, plus the cost due to system unavailability due to failures, plus the cost of recovering after each failure to the state given by the last checkpoint.

recent checkpoint is restarted. Thus, only the computations performed since the last checkpoint have to be repeated. Since the checkpoint operation has also an associated cost, the problem is to estimate the time between checkpoints such that the overall added cost (cost due to checkpoints and cost due to failures) is minimized.

Checkpointing is rarely used in Time Sharing systems except in programs where loss of data due to a failure is specially inconvenient (such as editors or electronic mail programs). However, it is extensively used in Real Time systems and in transaction processing systems, where at each checkpoint a copy of the database is made, and recovering from a failure means to bring the database to the last consistent state and reprocess the transactions arrived since the last checkpoint.

Because of its importance, the problem of determining the optimum checkpointing interval has received considerable attention. Table 7-2 is a summary of the most relevant models proposed for the evaluation of the optimum checkpointing interval. For each model a reference is given, the main assumptions in the model, and the decision criteria used to determine the optimum checkpointing interval. Most of these models have been surveyed in [Chandy 75b].

The purpose of this section is to investigate if the modeling methodology presented in this thesis confirms or invalidates the results given by the models presented in Table 7-2 and to study if a refining of these results is possible.

Reference	Assumptions	Goal
[Young 74]	· Constant workload · Constant failure rate · No errors during check.	Maximize Availability
[Chandy 75a]	· Constant workload · Constant failure rate · Errors occur during check.	Maximize Availability
[Chandy 75b]	· Periodic Workload · Periodic failure rate	Maximize number of transaction processed
[Gelenbe 78]	· Constant workload · Constant failure rate · Errors occur during check.	Minimize response time

Table 7-2: Four proposed models to evaluate the optimum checkpointing interval in a transaction processing system

7.3.1. Constant workload

If the workload is constant the failure process becomes a renewal process. The times between successive failures form a sequence of independent identically distributed random variables. Following the same approach as in Section 7.2, the added cost due to unreliability per unit time will be evaluated. If the checkpointing interval is assumed to be T_{ck}, the added cost is given by

$$E[C^{T_{ck}}] = \frac{1}{T_{ck}} \left[C_d + E[C^R|T_{ck}] \right] \tag{7.34}$$

$E[C^R|T_{ck}]$ is the expected cost dut to recoveries from possible failures given that the checkpoint interval is T_{ck}. Repeating the same reasoning as in Section 7.2,

$$E[C^R|T_{ck}] = \sum_{n=1}^{\infty} P_{T_{ck}}(N_f = n)\, E[C^R|T_{ck}, N_f = n] \tag{7.35}$$

$$= E[N_f|T_{ck}] \left[C^{R_0} + k\frac{T_{ck}}{2} \right] \tag{7.36}$$

where it has been assumed that the recovery cost after each failure is equal to a fixed cost C^{R_0} plus a variable cost proportional to the time since the last checkpoint. The expected variable cost is $kT_{ck}/2$. Also, for a renewal process, the expected number of failure during the time T_{ck} is T_{ck} divided by the Mean Time To Failure (MTTF). Hence,

$$E[C^R|T_{ck}] = \frac{T_{ck}}{MTTF}\left[C^{R0} + k\frac{T_{ck}}{2}\right] \tag{7.37}$$

and

$$E[C^{T_{ck}}] = \frac{1}{T_{ck}}\left[C_d + \frac{T_{ck}}{MTTF}\left(C^{R0} + k\frac{T_{ck}}{2}\right)\right] \tag{7.38}$$

The expected cost will be minimum when its derivative with respect to T_{ck} is zero. The optimum checkpointing interval is therefore given by

$$T_{ck} = \left(\frac{2\,C_d\,MTTF}{k}\right)^{1/2} \tag{7.39}$$

which is exactly the result obtained by [Young 74].

Indeed, since the failure process is a renewal process, the expected cost depends on the expected number of renewals (failures) per unit time, but it is independent of the PDF of the time to failure. Therefore any result obtained under the assumptions of this thesis will agree with previously obtained results if the average workload is assumed to be constant.

7.3.2. Periodic workload

If the average system workload is given by w(t) and the average failure rate is $\lambda(t)$ [Chandy 75a] has given a recursive algorithm to determine the optimum sequence of checkpoint times to minimize the added cost due to unreliability. The solution given by [Chandy 75a] is based on discretizing m(t) and $\lambda(t)$ in intervals during which they can be assumed to be constant. Graph theory can then be used to determine the optimum collection of checkpoint times.

The problem was originally stated by [Chandy 75a] as follows. If the last checkpoint was performed at time T_{ck}, the cost due to a possible failure at time t is

$$C^{[T_{ck},t]}_{t_f=t} = C^{R0} + k\int_{T_{ck}}^{t} w(s)\,ds \tag{7.40}$$

If the time required to perform a checkpoint is C_d, the total expected cost in $[T_{ck},t]$ is

$$E[C^{[T_{ck},t]}] = \int_{T_{ck}}^{T_{ck}+C_d} w(t)\,dt + \int_{T_{cd}+C_d}^{t} C^{[T_{ck},\tau]}_{t_f=t}\,\lambda(\tau)\,d\tau \tag{7.41}$$

Although according to the results presented in Chapter 4 $\lambda_t = m(t) + x_t$, the expected cost is equal to (7.41) since $E[\lambda_t] = m(t)$. The optimum checkpointing interval is the interval which minimizes the above cost. By discretizing m(t) [Chandy 75a] gives a recursive algorithm to compute the instants at which checkpoints must be done. The way in which the problem is stated is precisely the main

obstacle to obtain a concise solution. Instead, assume that the system is started at time t_s due to a failure or that a checkpoint finalized at time t_s. Then

$$E[C^{[t_s,T_{ck}]}] = k' \int_{t_s}^{T_{ck}} C^{[t_s,\tau]}_{t_f=\tau} \lambda(\tau)\, d\tau + \int_{T_{ck}}^{T_{ck}+C_d} w(t)\, dt \tag{7.42}$$

Differentiating with respect to T_{ck}, the following equation is obtained.

$$k'C^{R_0} + k'k\, \lambda(T_{ck}) \int_{t_s}^{T_{ck}} w(s)\, ds = w(T_{ck}) \cdot w(T_{ck}-C_d) \tag{7.43}$$

The difference between the value of T_{ck} satisfying (7.43) and the solution proposed by [Chandy 75a] is that the value of T_{ck} satisfying (7.43) can be computed by the system "on the fly". The first term on the left hand of (7.43) is the fixed cost due to recovery from a crash. The second term is the variable cost and increases as $T_{ck}-t_s$ increases. The right hand side is the cost associated with the checkpointing operation. Thus, the above equation indicates that checkpointing must be performed when the expected recovery cost exceeds the cost associated with checkpointing. Let the system be sampling the values of w(t) and $\lambda(t)$ at regular intervals Δt. Then,

$$\int_{t_s}^{t} W(s)\, ds = \sum_{n=1}^{N} w(t_n)\, \Delta t \tag{7.44}$$

where the first sample is taken immediately after a checkpoint has been performed or a crash has occurred. The system has only to keep track of the variables

$$C_R(t_n) = C_R(t_{n-1}) + k\, k'\, \lambda(t_n)\, w(t_n)\, \Delta t \tag{7.45}$$

$$C_{ck}(t_n) = w(t_n) \cdot w(t_n - C_d) \tag{7.46}$$

where

$$C_R = k'\, C^{R_0} \tag{7.47}$$

A checkpoint must be performed whenever $C_R(t_n) > C_{ck}(t_n)$. In this way, the optimum checkpointing interval adapts itself to system behavior, by resetting the time scale every time that a checkpoint or a crash occurs.

7.4. Reliability modeling including transient hardware faults, software faults, and permanent hardware faults

As a final elaboration of the present modeling methods, a model will be presented which includes the effect of permanent hardware faults, in addition to hardware transients and software faults. The modeled system is a nonredundant system under constant average workload. Recall from Chapters 5 and 6 that the Stationary model still gives a much better characterization than the any other model, even for systems under periodic workload. The assumptions regarding the statistics of the time to permanent fault will be the traditional ones, i.e., the time to permanent failure will be assumed to be exponentially distributed

$$P_p(t_p \leq \tau) = 1 - e^{-\lambda_p \tau} \tag{7.48}$$

where $P_p(t_p \leq \tau)$ is the probability that a permanent fault will occur before time τ. The PDF of the time to failure due to transients and software will be assumed to be any of the distributions given in Chapter 4 under the constant workload assumption

$$P_p(t_t \leq \tau) = 1 - e^{-\int_0^{\tau} h(s)\,ds} \tag{7.49}$$

where h(t) is any of the hazard functions given in Section 4.1.

7.4.1. Markov processes

Reliability modeling for permanent faults is often characterized by means of Markov processes. Central to the theory of Markov processes are the concepts of *state* and *state transition*. The state of a system represents all that it is needed to know to describe the system at any instant. In the course of time the system passes from state to state and therefore exhibits a dynamic behavior. If the system can be characterized by its continuous time evolution thorough a discrete state space, at any instant the system is in one of N states, and transitions between states occur at random times. The distinguishing property of Markov processes is that they must satisfy the following property

$$P(s_{t_n} = s_n | s_{t_1} = s_1, \ldots, s_{t_{n-1}} = s_{n-1}) = P(s_{t_n} | s_{t_{n-1}} = s_{n-1}) \tag{7.50}$$

where s_{t_n} denotes the state occupied at time t_n. The above equality has the following implications:

- The probability of occupying any state in the future depends only on the state presently occupied.
- The pdf of the time to the next transition does not depend on how long the present state has been occupied nor on the destination state

For continuous time Markov processes, the above property in fact implies that the time to transition must be exponentially distributed [Howard 71]. A stationary Markov process is then completely specified by its transition probability matrix

$$P = \{p_{ij}; i,j = 1,...,N\} \tag{7.51}$$

where

$$p_{ij} = P\{\text{next state is } j | \text{present state is } i\} \tag{7.52}$$

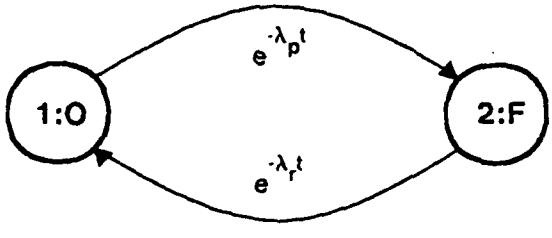
An equivalent characterization of a Markov process is in terms of the transition rates matrix Λ

$$\Lambda = \{\lambda_{ij}; i,j = 1,...,N\} \tag{7.53}$$

where

$$g_{ij}(t) = \lambda_{ij} e^{-\lambda_{ij} t} \tag{7.54}$$

is the pdf of the time to transition to state k given that the process enters state i at time 0.



State :	1:O	Operational
	2:F	Failed

Figure 7-6: Characterization of the reliability of a nonredundant system subject to permanent hardware faults by a Markov process. λ_p is the rate at which permanent failures occur and λ_r is the rate at which repairs take place.

Figure 7-6 summarizes the characterization of a nonredundant system subject to permanent hardware faults only. Since the system can be only operational or failed, the failure process is characterized as a 2 state Markov process. The times to failure and to repair are exponentially distributed. The MTTF and MTTR values are $1/\lambda_p$ and $1/\lambda_r$ respectively.

7.4.2. Semi-Markov processes

Markov processes are not appropriate to characterize the reliability of systems subject to permanent, transient, and software failures. The PDF of the time to failure due to transients or software has been shown to have a decreasing hazard function, and according to the statistical tests performed in Chaper 6, it is not properly described by an exponential distribution. Hence, for a system subject to the three types of failures the PDF of the time to transition depends on the destination state. This PDF will be exponential if the destination state is *failed due to a permanent failure*, or it will be of the form given in (7.49) if the destination state is *failed due to a transient or software error.*

This dependency of the pdf of the time to transition on the destination state is precisely the distinguishing property of the so called *Semi-Markov* processes. A system characterized by a Semi-Markov process is always in one of N states. Successive state occupancies are governed by the transition probabilities of a Markov process. At transition instants, the system behaves as a Markov process, and the process determining such transitions process is called the *embedded Markov process*. The imbedded Markov process is completely described by a NxN matrix of transition probabilities P as defined in (7.51). In addition, in a Semi-Markov process, whenever the system enters state i it is imagined that it determines the next state j according to state i's transition probabilities $\{p_{i1},...,p_{iN}\}$. After j has been chosen, the system "holds" for a random time τ_{ij} in state i. The pdf of τ_{ij} is given by $q_{ij}(t)$, obtaining therefore a vector of pdf's for each state i. Hence, a Semi-Markov process is completely determined only if both the matrix P and the pdf's matrix Q(t)

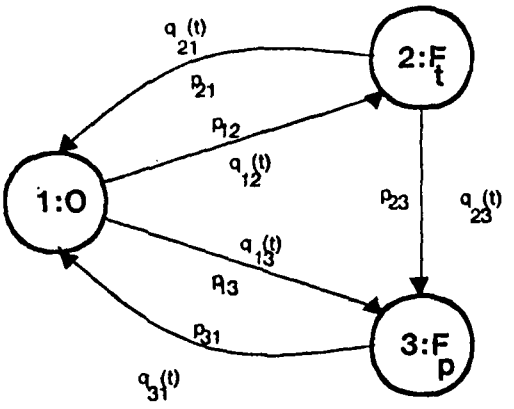
$$Q(t) = \{q_{ij}(t);i,j = 1,...,N\} \tag{7.55}$$

are available.

Figure 7-7 synthizes how a non redundant computing system can be characterized by a Semi-Markov process incorporating the effects of permanent hardware failures, transient hardware failures, and software failures. The system is operational when in state 1. The system selects then the next state according to the transition probabilities p_{12}, p_{13}. If the destination state is state 2 (Failed due to transients or software), the system selects

$$q_{12}(t) = h(t)\, e^{-\int_0^t h(s)\,ds} \tag{7.56}$$

as the pdf of the time to transition. If the next state is 3 (failed due to a permanent hardware failure) an exponential distribution with parameter λ_p is selected as the PDF of the time to transition



State :	1:O	Operational
	2:F_t	Failed due to transients or software
	3:F_p	Failed due to permanent failures

Figure 7-7: Characterization of a non redundant system subject to permanent and transient hardware failures, and software failures

$$q_{13}(t) = \lambda_p e^{-\lambda_p t} \tag{7.57}$$

The other transitions are similarly characterized. If the system is in state 2 (failed due to transients or hardware) it will become operational after a fixed recovery time t_r and therefore the pdf of the time to restart is $q_{21}(t) = \delta(t_r)$. A permanent hardware failure may also occur while the system is recovering from a transient. The pdf of the time to such event is an exponential distribution truncated at $t = t_r$. If the system is in state 3 (failed due to a permanent hardware failure) it will always recover after a random time exponentially distributed with parameter λ_r and

$$q_{31}(t) = \lambda_r e^{-\lambda_r t} \tag{7.58}$$

Note that λ_r is not the rate at which permanent failures *occur*, but the rate at which permanent failures are observed *since the last system restart*.

7.4.2.1. Limiting behavior

Define now the following matrix of time varying functions

$$\Phi(t) = \{\varphi_{ij}(t);i,j = 1,...,N\} \tag{7.59}$$

where $\varphi_{ij}(t)$ is the probability that the system *will occupy state j at time t given* that it *entered* state i at time 0. Then it can be proven that

$$\varphi_{ij}(t) = \delta_{ij} \int_t^{\infty} q_i(\tau)\, d\tau + \sum_{k=1}^{N} p_{ik} \int_0^t q_{ij}(\tau)\, \varphi_{ij}(t-\tau)\, d\tau \tag{7.60}$$

Equation (7.60) requires the solution of a system of integral equations whenever numerical values of $\Phi(t)$ are required. Although this system of equations can sometimes be solved by using Laplace transform methods (see [Howard 71]) it is cumbersome. However, if the desired knowledge is only *on the average*, a simpler result can be used. Let

$$\varphi_{ij} = \lim_{t\to\infty} \varphi_{ij}(t) \tag{7.61}$$

Then φ_{ij} is the average fraction of time spent in time j given that the system entered state i at time 0. For example, the value of φ_{11} for the system described in Figure 7-7 is the system availability, since it is equal to the fraction of time that the system is operational given that the system was first started at time 0. A basic result of Semi-Markov theory is

$$\varphi_{ij} = \frac{\pi_j E[\tau_j]}{\sum_{i=1}^{N} \pi_j E[\tau_j]} \tag{7.62}$$

$$= \frac{\pi_j E[\tau_j]}{E[\tau]} \tag{7.63}$$

where $\vec{\pi} = (\pi_1,...,\pi_N)$ is the limiting state probability vector of the embedded Markov process. Such a vector can be obtained by solving the system of equations

$$\vec{\pi} = \vec{\pi} P \tag{7.64}$$

subject to the condition

$$\sum_{i=1}^{N} \pi_i = 1 \tag{7.65}$$

Equation (7.63) is important in that it implies that the only statistic of the holding times that affects the limiting behavior of states occupancies is the expected value.

As a simple example consider the CMU-10A. Repair takes place with a frequency smaller than once a month. Since the system crashes on the average every 9 hours,

$$p_{12} = 0.999 \tag{7.66}$$

$$p_{13} = 0.001 \tag{7.67}$$

Also, it will be assumed that

$$p_{21} = 1 \tag{7.68}$$

$$p_{23} = 0 \tag{7.69}$$

$$p_{31} = 1 \tag{7.70}$$

$$p_{32} = 0 \tag{7.71}$$

The following values are then obtained for $\vec{\pi}$

$$\pi_1 = 0.5 \tag{7.72}$$

$$\pi_2 = 0.4995 \tag{7.73}$$

$$\pi_3 = 0.0005 \tag{7.74}$$

Assuming

$$E[\tau_1] = 9 \text{ hours (Mean Time To Failure)} \tag{7.75}$$

$$E[\tau_2] = 15 \text{ minutes (Mean time to recover from transients)} \tag{7.76}$$

$$E[\tau_3] = 2 \text{ hours (Mean Time To Repair)} \tag{7.77}$$

then

$$\text{Availability} = \varphi_{11} = 0.97 \tag{7.78}$$

7.4.2.2. Reliability prediction

Let the pdf of the time to failure, $p_{tf}(t)$ be the unconditional pdf of the time to transition from state 1, independently of the destination state. Thus,

$$p_{tf}(t) = p_{12}\, q_{12}(t) + p_{13}\, q_{13}(t) \tag{7.79}$$

and the reliability function becomes

$$R(t) = P(t_f \geq t) \tag{7.80}$$

$$= \int_t^{\infty} p_{tf}(s)\, ds \tag{7.81}$$

$$= p_{12}\, R_{12}(t) + p_{13}\, R_{13}(t) \tag{7.82}$$

where $R_{12}(t)$ is the reliability function due to transients and software

$$R_{12} = e^{-\int_0^t h(s)\, ds} \tag{7.83}$$

and R_{13} is the reliability function due to permanent failures

$$R_{13}(t) = e^{-\lambda_p t} \tag{7.84}$$

Since $p_{12} \ggg p_{13}$, the system reliability is essentially the reliability function discussed in Chapter 6 for transients and software according to the Stationary model.

7.5. Summary

The modeling methodology introduced in Chapters 3 and 4 has been extended in this chapter to derive some important applications to Reliability modeling and cost/benefit analysis of fault-tolerance. In particular, the following extensions have been considered:

- Decomposition of the failure process in its software and hardware components. Although on the average the probability that a crash is due to software may be of 0.6, the impact of unreliable software may be much more important due to the fact that the system crashes more often in periods of high load when the contribution of uncorrect software is larger than average.

- Evaluation of the added cost due to unreliability both from a user's viewpoint and from a system manager's viewpoint. Curiously, the fact of having a periodic workload (as opposed to constant) has an associated cost in itself.

- Study of previous results to evaluate the optimum checkpointing interval. A new result has been presented in the case of periodic workload.

- Introduction to models incorporating the effects of software errors, transient hardware faults, and permanent hardware faults.

Chapter 8
Conclusions and suggestions for further research

Through the thesis, the two main questions for which a simultaneous answer has been sought are

Question 1 : What is it desirable to know about computing systems reliability?

Question 2 : What variables can be easily measured from real systems?

If a simultaneous answer for both questions exists, it must obviously a compromise, since the answer to the first question is "everything". And this will never be known. Prehaps the closest answer to the above two questions are the results presented in Chapter 7, were methods to evaluate the impact of unreliability, and methods to trace the impact of each cause of unreliability (permanent hardware failures, transient hardware failures, and software failures) have been presented.

It was in Chapter 2 that it was claimed that an apparent conflict would be solved. The fact that a system fails more during prime time is widely accepted. And no statistical tests can contradict the fact that the Weiubull distribution characterizes is a better distribution to characterize the time to failure than an Exponential or Periodic model, even though the Weibull model does not include periodicity concepts. The answer to this apparent conflict seems to be to consider the failure rate to be a Gaussian process with periodic statistics, i.e., a cyclostationary process. The after effects of this approach have been

- Derivation of the general properties of the class of Doubly Stochastic Poisson process whose failure rate is a Gaussian process (Chapter 3).
- Characterization of doubly stochastic Poisson process whose intensity is either a stationary or a cyclostationary Gaussian process. In particular a complete family of distributions commonly used in statistical analysis of failure date have been shown to be special cases of this approach (Chapter 4). As a side effect, the general properties of the unreliable behavior of computing systems operating under periodic or constant workload have been established.
- Elaboration of the necessary techniques for model parameterization and validation (Chapter 5).

- Validation of the model by comparison with the actual behavior of a real system, and comparison of its predictions with the predictions of other models (Chapter 6). In particular, stablishment of ranges for which other models may lead to over optimistic or over pessimistic expectations.

- Establishment of cost and benefit measures of fault-tolerance derived from the modeling methodology (Chapter 7).

Since the main results presented in Chapters 3,4, and 7 are original, a cautious approach must be taken in deriving conclusions from these results until further proofs of their validity are available. With caution and the two above questions in mind, the following sections summarize the preliminary conclusions derived from this thesis and pose some interesting unanswered questions. Traditionally, these new questions will require some more research to be answered, or they will be forgotten.

8.1. Reliability modeling

Through the thesis a reliability modeling methodology has been developed starting from basic principles of operation of Time Sharing systems. Nevertheless, it should be noted that the original MULTICS design dates from the early 1960's. Model validation has been done with the TOPS-10 operating system, already more than 10 years old. Why then bother to study such systems? Would not it be better to study state of the art Time Sharing systems, multiprocessors, multicopmuters, or collections of personal computers operating in Local Area Networks ?

The fact is that current systems still adopt the basic conventions of the original MULTICS design. For example, the IBM 4341 processor executing the operating system VM/370 does not attempt to recover from transient hardware failures if these failures occur while the system executes in kernel mode [Ciacelly 81]. The system attempts to recover when transient failures occur in other modes, which is one of the central hypothesis of work of the present thesis. As for the extension of similar modeling methods to other systems such as multiprocessors or multicomputers, note that the methods followed in this thesis are oriented to the *steady state* system characterization relying heavily on operating system measuring system facilities such as error logs, system tables with accounting information, and so on. These measuring tools are available in operating systems for purposes of accounting, maintenance aids, or system tunning facilities. But these measuring facilities have been used here for reliability characterization purposes.

Few multiprocessors are available today for general use and experimentation. Of the

multiprocessors available, most are experimental systems (such as C.mmp [Wulf 81] and Cm* [Jones 80]) are far from being general purpose systems or are dedicated to the execution of relatively simple real time functions (such as Pluribus [Ornstein 74] whose only function is that of packet switching, or other multiprocessors dedicated to telephone switching functions). Obviously, in most experimental systems the concept of steady state operation is not defined, and being vehicles of experimentation, the software is usually changing continuously. Further, no system available for experimentation has the necessary measuring tools required to validate theoretical models. The emphasis given in Tandem systems [Katzmanu 77] to instrumentation problems is significant [Blake 80], since Tandem systems are at present the only multiprocessors offering high reliability in general purpose applications. Further, before attacking the problem of characterizing the unreliable behavior of multiprocessors due to hardware transients and software, it seems reasonable to solve first the problem for simpler, more accessible systems such as Time Sharing computers.

Nevertheless, it is expected that several of the new results presented in this thesis will be applicable to other systems. In Local Area Networks, expensive facilities such as centralized file systems or expensive peripherals are likely to operate in Time Sharing mode, their reliability characterization being characterized by the same principles exposed in this thesis.

The model presented in Section 7.4 incorporating permanent hardware failures, transient failures, and software failures can be viewed as a first step in the characterization of the unreliable behavior of multiprocessor systems. The extension of this model to multiprocessors is desirable but not at all an easy task. First, note that model parameterization is possible only after detailed knowledge about the relationships between resource usage and unreliable manifestations. Remember how the PDF of the time to failure due to hardware transients and software has been derived. Secondly, the introduction of redundancy in hardware, software, or both may lead to unexpected results since the failure processes due to software and transients are *not* independent, but both depend on workload time varying patterns. Thus, care is necessary when elaborating the model to systems with some degree of redundancy.

A problem that has been systematically ignored through the thesis is the distinction between transients and *intermittent* faults. While transient failures are manifestation of changing environmental conditions (such as cosmic rays) or consequences of limitations in manufacturing processes (such as the presence of radioactive materials in packaging materials), intermittent faults are manifestations of physical degenerative processes (for instance, oxidation in a terminal contact). Still, much of the results presented in this thesis should be valid, since an intermittent fault can only

be detected when exercising the component affected by the degenerative process. However, the distinction between transients and intermittents is useful for diagnosis and replacement policies. Current efforts in this direction have been reported by [Bossen 81].

Finally, a sensitivity analysis should be performed establishing levels of confidence of the reliability predictions of the Stationary and Cyclostationary models depending on the parameter estimation procedures and sample size.

In summary, the main topics in which further research is to be expected are :

- Incorporation in state of the art systems of exhaustive measuring capabilities to allow system characterization and model validation with relatively minor effort.
- Extension of the present modeling methods (i.e., hardware/software prediction models) to systems having some degree of redundancy at the subsystem level such as multiprocessors.
- Better understanding in the differences in the manifestations of transients and intermittent faults.
- Sensitivity analysis of reliability predictions.

8.2. Performance/Reliability modeling

The above considerations are specially relevant to Performance/Reliability modeling techniques of systems having some degree of redundancy such as multiprocessors. While in a (uniprocessor) Time Sharing system singularities are easily identified (i.e., the hardware and the kernel of the operating system) for a multiprocessor singularities may form a dynamically changing collection of resources.

Some Performance/Reliability models were referenced in Chapter 2. Most of those models assume that upon failure detection the system may reconfigure itself and continue operating in a degraded performance state until repair takes place. These models attempt then to characterize how system performance is likely to evolve in time depending on the presence of different types of failures. The main assumption common to all these existing models is that they all use Markov models as the underlying abstraction. That means that all the times between state transitions are exponentially distributed. However, it has been shown in this thesis that the distribution of the time to failure due to transients and software cannot be approximated by an exponential distribution. Therefore, Performance/Reliability models will have to evolve into Semi-Markov models where the distributions of failures due to software and transients are of the type derived in Chapter 4.

If all it is needed to know about the system is its steady state behavior, the distinction between Markov and Semi-Markov modeling is not important. As it has been shown in Section 7.4 steady state characterization for a Semi-Markov model depends only on the expected times between transitions. However, if some more detailed knowledge is required, Semi-Markov modeling is unavoidable. Recall from Section 6.1.2 that the differences in reliability predictions between the exponential distribution and the distribution predicted by the Stationary approximation are not neglectable for values of time smaller than the MTTF value. Therefore, the distiction between Markov and Semi-Markov modeling is especially relevant for systems having exceptional reliability requirements during periods of time smaller than the expected MTTF value. This is the case, for example, of SIFT [Wensley 78] and FTMP [Hopkins 78].

8.3. Software reliability evaluation and the design of reliable software

The central argument of this thesis with respect to software reliability is that the observed software reliability depends on the instantaneous complexity of the data to be processed. Certainly, when a software package is implemented it is expected to cope equally well in all situations for which it has been designed to work. However, given that the software is subject to imperfections, it is more likely the such imperfections will be noticed while processing data describing situations of high complexity than processing data describing simple situations. This is so because simpler situations are easier to understand, the software for them is easier to design, and easier to debug.

This discussion is in rather loose terms because the lack of a suitable descriptor for the meaning of "complexity". But note that here complexity is an attribute of the world as seen by the software, not an attribute of the software itself. However, the world seen by the software is just the state of its data structures. If the only descriptors that can be obtained about the complexity of a situation to be processed by the software is by means of the state of its data structures, such descriptors will be very much representation dependent. By dependency on representation it is meant that different situations with the same inherent complexity may lead to different software reliability characterizations depending on the representation adopted in the data structures to represent such complexity. Consider, for instance, the problem of deadlocks. Several processes request the allocation of several resources. If some processes are processing for each other's resources but all are waiting and none is able to release a resource, deadlock occurs. However note that the number of processes, requests, and resources (which together determine the complexity of the situation to be processed) deadlock

will occur only under certain ordering of requests, while sequences with different orderings may be handled properly. The argument now is that *on the average* data describing more complex situations will be harder to process, and therefore more prone to the manifestation of a software fault.

This representation dependency of the complexity of the world seen by the software, and its relevance to software reliability manifestations has however some potential advantages. Consider a piece of code that operates on certain data structures in such a way that, the same code, fed with the same input data, uses *different* internal representations in its data structures according to some random factor. Then code replication to increase reliability makes sense!

Indeed, consider a software package operating over a variety of data structures such as lists, queues, and arrays. Assume that the code has been written in such a way that data structure initialization (and even perhaps allocation) is random. That is, no two initialization sequences lead to the same representation of the same situation. This can be accomplished, for instance, by chosing the header for the queues at random in a circular buffer. Consider now two copies of the same code running in parallel in separate processors (or sequentially in the same processor). As the code is feeded with external input data, both copies will use different representations in their internal data structures. Thus, in some cases, one copy may manifest a software fault due to a particular representation, while the other copy may be able of handling the same situation without problem.

The above arguments are highly speculative and their validation would require (at least) the design of a complete experiment and background study as it has been done in the present thesis. However, this potential approach to the design of reliable software has been presented here because it is a natural extension of the methodology followed in this thesis.

References

[Anderson 67] J.E. Anderson and F.J. Macri.
Multiple Redundancy Applications in a computer.
In *Proceedings of the 1967 Annual Symposioum on Reliability*, pages 553-562. January, 1967.

[Anscombe 52] F.J. Anscombe.
Large sample theory of sequential estimation.
Proc. Camb. Phil. Soc. 48:600-607, 1952.

[Avizienis 75] A. Avizienis.
Fault-Tolerance and Fault-Intolerance: Complementary Approaches to Reliable Computing.
In *Proc., 1975 Int. Conf. Reliable Software*, pages 458-464. IEEE Comp. Soc., April, 1975.

[Avizienis 77] A. Avizienis and L. Chen.
On the Implemenation of N-Version Programming for Software Fault-Tolerance During Execution.
In *Proc., COMPSAC 77*, pages 149-155. IEEE Comp. Soc., November, 1977.

[Barlow 75] R. E. Barlow and F. Proschan.
Statistical Theory of Reliability and Life Testing: Probability Models.
Holt, Rinehart, and Winston, Inc., 1975.

[Beaudry 78] M.D. Beaudry.
Performance Related Reliability Measures for Computing Systems.
IEEE Trans. Computers C-27(6):540-547, June, 1978.

[Beaudry 79] M. D. Beaudry.
A Statistical Analysis of Failures in the SLAC Computing Center.
In *Digest of Papers, COMPCON Spring 79*, pages 49-52. IEEE Comp. Soc., 1979.

[Beaudry 80] M.D. Beaudry.
Stochastic Behavior of Failures in Computing Systems.
Technical Report 172, Center for Reliable Computing, Stanford University, February, 1980.

[Bell 78] C.G. Bell, A. Kotok, T.N Hastings, and R. Hill.
The Evolution of the DECsystem 10.
Communications of the Association for the Computing Machinery 21(1):44-63, January, 1978.

[Berger 74] R.W. Berger and K. Lawrence.
Estimating Weibull Parameters by Linear and Nonlinear Regression.
Technometrics 16(4):617-619, November, 1974.

[Billingsley 56] P. Billingsley.
The invariance principle for dependent random variables.
Trans. Amer. Math. Soc. 83:250-268, 1956.

[Billingsley 63] P. Billingsley.
Limit theorems for randomly selected partial sums.
Ann. Math. Stat. 33:85-92, 1963.

[Billingsley 79] P. Billingsley.
Probability and Measure.
Wiley, 1979.

[Blake 80] R. Blake.
Xray: Instrumentation for Multiple Computers.
In *Proc. Performance 80*, pages 11-25. ACM Sigmetrics 9(2), May, 1980.

[Bossen 81] D.C. Bossen and M.Y. Hsiao.
ED/FI: A Technique for Improving Computer System RAS.
In *Proc. FTCS-11*, pages 2-7. IEEE Comp. Soc., June, 1981.

[Bouricious 69] W.G. Bouricious, W.C. Carter, and P.R. Schneider.
Reliability Modeling Techniques for self-repairing Computer Systems.
In *Proceedings of the 24th National Conference of ACM*, pages 295-383. 1969.

[Breiman 68] L. Breiman.
Probability.
Addison Wesley, 1968.

[Butner 80] S.E. Butner and R.K. Iyer.
A Statistical Study of Reliability and System Load at SLAC.
Technical Report, Center for Reliable Computing, Stanford University, January, 1980.

[Castillo 80a] X. Castillo, D.P. Siewiorek.
A Performance-Reliability Model For Computing Systems.
Technical Report, Carnegie-Mellon University, Computer Science Department, 1980.

[Castillo 80b] X. Castillo.
Workload, Performance, and Reliability of Digital Computing Systems.
Technical Report, Carnegie-Mellon University, Computer Science Department, December, 1980.

[Chandy 75a] K.M. Chandy, J.C. Browne, C.D. Dissly, and W.R. Uhrig.
Analytic Models for Rollback and Recovery Strategies in Data Base Systems.
IEEE Trans. Soft. Eng. SE-1(1):100-110, March, 1975.

[Chandy 75b] K.M. Chandy.
A survey of Analytic Models of Rollback and Recovery Strategies.
Computer 8(5):40-47, May, 1975.

[Cheung 75] R.C. Cheung and C.V. Ramamoorthy.
Optimal Measurement of Program Path Frequencies and its Applications.
In *Proc. 1975 Int. Fed. Automat. Contr. Congr.*. August, 1975.

[Cheung 80] R.C. Cheung.
A User-Oriented Software Reliability Model.
IEEE Trans. Software Engineering SE-6(6):118-125, March, 1980.

[Chou 80] T.C.K. Chou and J.A. Abraham.
Performance/Availability model of Shared Resource Multiprocessors.
IEEE Trans. Reliability R-29(1):70-74, April, 1980.

[Ciacelly 81] M.L. Ciacelly.
Fault-Handling of the IBM 4341 Processor.
In *Proc. FTCS-11*, pages 9-12. IEEE Comp. Soc., June, 1981.

[Cinlar 72] E. Cinlar.
Superposition of point processes.
In P.A.W. Lewis (editor), *Stochastic Point Processes Statistical Analysis, Theory, and Applications*, pages 549-606. Wiley, 1972.

[Corbato 74] F.J. Corbato, J.H. Saltzer, and C.T. Clingen.
MULTICS-The first seven years.
In *AFIPS Conf. Proceedings*, pages 571-583. 1974.

[Costes 78] A. Costes, C. Landrault, and J.C. Laprie.
Reliability and Availability Model for Maintained Systems Featuring Hardware Failures and Design Faults.
IEEE Trans. Computers C-27(6):548-560, June, 1978.

[Digital 77] *TOPS-10 Monitor Calls Manual*
Digital Equipment Corporation, 1977.

[Digital 78] *TOPS-10 and TOPS-20 SYSERR Manual*
Digital Equipment Corporation, 1978.

[Ferdinand 74] A.E. Ferdinand.
A Theory of Systems Complexity.
Int. J. Gen. Syst. 1:19-33, 1974.

[Fitzsimmons 78] A. Fitzsimmons and T. Love.
A review and evaluation of Software Science.
ACM Computing Surveys 10(1):3-18, March, 1978.

[Fuller 78] S.H. Fuller and S.P. Harbison.
The C.mmp multiprocessor.
Technical Report, Carnegie-Mellon University, Computer Science Department, October,, 1978.

[Gardner 75] W. A. Gardner, L. E. Franks.
Characterization of Cyclostationary Random Signal Processes.
IEEE Trans. Information Theory IT-21(1):4-14, January, Year = 1975.

[Gardner 78] W. A. Gardner.
Stationarizable Random Processes.
IEEE Trans. Information Theory IT-24(1):8-22, January, 1978.

[Gay 79] F. A. Gay and M. L. Ketelsen.
Performance Evaluation for Gracefully Degrading Systems.
In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 51-58. IEEE Comp. Soc., 1979.

[Geilhufe 79] M. Geilhufe.
Soft errors in semiconductor memories.
In *Digest of Papers, COMPCON Spring 79*, pages 210-216. IEEE Comp. Soc., 1979.

[Gelenbe 78] E. Gelenbe and D. Derochette.
Performance of Rollback Recovery Systems under Intermittent Failures.
ACM Comm. 21(6):493-499, June, 1978.

[Glass 81] R.L. Glass.
Persistent Software Errors.
IEEE Trans. Software Engineering SE-7(2):162-168, March, 1981.

[Goldstine 72] H.H. Goldstine.
The Computer from Pascal to Von Newmann.
Princeton University Press, 1972.

[Grnarov 80] A. Grnarov, J. Arlat, A. Avizienis.
On the Performance of Software Fault-Tolerance Strategies.
In *Proc. FTCS-10*, pages 251-253. IEEE Comp. Soc., October, 1980.

[Harris 68] C.M. Harris and N.D. Singpurwalla.
Life Distributions Derived from Stochastic Hazard Functions.
IEEE Trans. Reliability R-17(2):70-79, June, 1968.

[Hecht 76] H. Hecht.
Fault-Tolerant Software for Real-Time Applications.
ACM Computing Surveys 8(4):391-407, December, 1976.

[Hodges 77] D.A. Hodges.
Progress in Electronic Technologies for Computers.
Technical Report, National Bureau of Standards, March, 1977.

[Hopkins 78] A.L. Hopkins, T.B. Smith, and J.H. Lala.
FTMP-A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft.
IEEE Proc. 66(10):1221-1239, October, 1978.

[Horowitz 75] E. Horowitz.
Practical Strategies for Developing Large Scale Systems.
Addison-Wesley, 1975.

[Howard 71] R. A. Howard.
Dynamic Probabilistic Systems.
Wiley, 1971.

[Jelinsky 73] A. Jelinsky and P.B. Moranda.
Applications of a probability Based Method to a Code Reading Experiment.
In *Proceedings of the 1973 Symposioum on Software Reliability*, pages 78. IEEE Comp. Soc., 1973.

[Jenkins 68] G.M. Jenkins and D.G. Watts.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[Jones 80] A.K. Jones and E.F. Gehringer, Eds.
The Cm Multiprocessor Project: a Research Review.*
Technical Report CMU-CS-80-131, Carnegie-Mellon University, Computer Science Department, July, 1980.

[Katzmanu 77] J. A. Katzmanu.
A Fault-Tolerant Computing System.
Technical Report, Tandem Computers Inc., 1977.

[Keller 76] T.W. Keller.
CRAY-1 Evaluation Final Report.
Informal Report LA-6456-MS, Los Alamos Scientific Laboratory, December, 1976.

[Keyes 81] R.W. Keyes.
Fundamental Limits in Digital Information Processing.
IEEE Proc. 69(2):267-268, February, 1981.

[Kleinrock 75] L. Kleinrock.
Queuing Systems.
Wiley, 1975.

[Lee 79] P.A. Lee et al.
A Recovery Cache for the PDP-11.
In *Proc., 1979 Int. Symp. Fault-Tolerant Computing*, pages 3-8. IEEE Comp. Soc., June, 1979.

[Littlewood 79] B. Littlewood.
How to Measure Software Reliability and How Not To.
IEEE Trans. Reliability R-28(2):103-110, June, 1979.

[Lynch 75] W.C. lynch, W. Wagner, and M.S. Schwartz.
Reliability Experience with Chi/OS.
IEEE Trans. Software Engineering SE-1(2):253-257, June, 1975.

[May 79] T.C. May.
Soft errors in VLSI - Present and Future.
In *Proc., 29th Electronic Components Conference*, pages 247-256. IEEE, 1979.

[McConnel 79a] S. R. McConnel, D. P. Siewiorek, and M. M. Tsao.
Transient Error Data Analysis.
Technical Report CMU-CS-79-121, Carnegie-Mellon University, Departments of Electrical Engineering and Computer Science, May, 1979.

[McConnel 79b] S. R. McConnel, D. P. Siewiorek, and M. M. Tsao.
The Measurement and Analysis of Transient Errors in Digital Computing Systems.
In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 67-70. IEEE Comp. Soc., 1979.

[McConnel 81] S.R. McConnel.
Analysis and Modeling of Transient Errors in Digital Computers.
PhD thesis, Carnegie-Mellon University, June, 1981.

[Melsa 78] J.L. Melsa and D.L. Cohen.
Decision and Estimation Theory.
McGraw-Hill, 1978.

[Meyer 79] J. F. Meyer, D. G. Furchtgot, and L. T. Wu.
Performability evaluation of thh SIFT computer.
In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 43-50. IEEE Compter Society, 1979.

[Miyamoto 75] I. Miyamoto.
Software Reliability in Online Environment.
In *Digest of Papers, 1975 International Conference on Software Reliability*, pages 194-203. IEEE Comp. Soc., 1975.

[Mohanly 73] S.N. Mohanly.
Models and Measurements for Quality Assesment of Software.
ACM Computing Surveys 11(3):250-275, September, 1973.

[Moreira 80] J. Moreira de Souza.
A unified method for the benefit analysis of Fault-Tolerance.
In *Proc. FTCS-10*, pages 201-203. IEEE Comp. Soc., 1980.

[Morganti 78] M. Morganti, G. Coppadoro, and S. Ceru.
UDET 7116 - Common Control for PCM Telephone Exchange - Diagnostic Software Design and Availability Evaluthation.
In *Proc. FTCS-8*, pages 16-23. IEEE Comp. Soc., 1978.

[Musa 75] J.D. Musa.
A Theory of Software Reliability and its Applications.
IEEE Trans. on Software Engineering SE-1(1):312-327, September, 1975.

[Nelson 73] E.C. Nelson.
A Statistical Basis for Software Reliability Assesment.
Technical Report, TRW, March, 1973.

[Ohm 79] V.J. Ohm.
Reliability Considerations for Semiconductor Memories.
In *Digest of Papers, COMPCON Spring 79*, pages 207-209. IEEE Comp. Soc., 1979.

[Oppenheim 75] A.V. Oppenheim and R.W. Schafer.
Digital Signal Processing.
Prentice-Hall, 1975.

[Organick 72] E.I. Organick.
The MULTICS System - An examination of its structure.
MIT Press, 1972.

[Ornstein 74] S.M. Ornstein et al.
Pluribus - A Reliable Multiprocessor.
In *Proceedings of the 1974 Computer Conference*, pages 551-559. 1974.

[Papoulis 65] A. Papoulis.
Probability, Random Variables, and Stochastic Processes.
McGraw-Hill, 1965.

[Phister 79] M. Phister Jr.
Data Processing Technology and Economics.
Digital Press, 1979.

[Powell 78] M.J.D. Powell.
Algorithms for Nonlinear Constraints that use Lagrangian Functions.
Mathematical Programming 14(2):224-248, 1978.

[Randell 75] B. Randell.
System Structure for Software Fault-Tolerance.
IEEE Trans. Software Eng. SE-1(2):220-232, June, 1975.

[Randell 78] B. Randell, P.A. Lee, P.C. Treleaven.
Reliability Issues in Computing System Design.
ACM Computing Surveys 10(2):123-165, June, 1978.

[Reynolds 75] C.H. Reynolds and J.E. Kinsbergen.
Tracking Reliability and Availability.
Datamation 21(11):106-116, November, 1975.

[Romano 77] A. Romano.
Applied statistics for Science and Industry.
Allyn and Bacon Inc., 1977.

[Russell 78] R. M. Russell.
The CRAY-1 Computer System.
Communications of the Association for the Computing Machinery 21(1):63-72, January, 1978.

[Saleh 74] Saleh.
Probability Distribution of Time of Arrival of Photoevents for a Stationary Optical Field.
IEEE Trans. on Information Theory IT-20(2):262-263, March, 1974.

[Schick 78] G.J. Schick and R.W. Wolverton.
An Analysis of Compiting Software Reliability Models.
IEEE Trans. on Software Engineering SE-4(2):104-120, March, 1978.

[Sheppard 62] C.W. Sheppard.
BAsic Principles of the Tracer Method.
Wiley, 1962.

[Shooman 73] M.L. Shooman.
Operational Testing and Software Reliability Estimation During System Development.
In *Proceedings of the 1973 Symposioum on Computer Software Reliability*, pages 51-57. IEEE Comp. Soc., April, 1973.

[Siewiorek 78] D.P. Siewiorek, V. Kini, H. Mashburn, S.R.McConnel, M.M.Tsao.
A case study of C.mmp, Cm*, and C.vmp - Part I - Experiences with Fault-Tolerance in Multiprocessor Systems.
IEEE Proc. 66(10):1178-1199, October, 1978.

[Siewiorek 80] Dan Siewiorek and Dave Rennels.
Workshop Report - Fault-Tolerant VLSI Design.
Computer 13(12):51-53, December, 1980.

[Smith 81] A.L. Smith.
Hard and Soft Failures in Dynamic RAM Fault Tolerant Memories.
IEEE Trans. Reliability R-30(1):58-60, April, 1981.

[Snyder 75] D. L. Snyder.
Random Point Processes.
John Wiley & Sons, 1975.

[SRC 81] Software Reliability Committee.
AdCom Committee Reports.
IEEE Reliability Society Newsletter 27(2):7, April, 1981.

[Stratonovich 67] R. L. Stratonovich.
Topics in the Theory of Random Noise, Vol. II.
Gordon and Breach, 1967.

[Thayer 78] T.A. Thayer, M. Lipow, and E.C. Nelson.
Software Reliability.
North-Holland Publishing Co., 1978.

[Thoman 69] D.R. Thoman, L.J. Bain, and C.E. Antle.
Inferences on the Parameters of the Weibull Distribution.
Technometrics 11(3):445-460, August, 1969.

[Turnbull 74] B.W. Turnbull, B.W. Brown Jr., M. Hu.
Survivorship analysis of heart transplant data.
J. Amer. Statist. Assoc. 69:74-80, March, 1974.

[Von Newmann 63]
John Von Newmann.
Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components.
In A.H. Taub (editor), *Collected Works*, pages 329-378. Pergamon Press, 1963.

[Weibull 51] W. Weibull.
A distribution of wide applicability.
J. Appl. Mech. 18(3):293-297, 1951.

[Wensley 78] J.H. Wensley et al.
SIFT: The design and Analysis of a Fault-Tolerant Computer for Aircraft Control.
IEEE Proc. 66(10):1240-1254, October, 1978.

[Wong 79] E. Wong.
Stochastic Processes in Information and Dynamical Systems.
Krieger, 1979.

[Wulf 81] W.A. Wulf, R. Levin, and S.P. Harbison.
HYDRA/C.mmp - An Experimental Computer System.
McGraw-Hill, 1981.

[Young 74] J.W. Young.
A First Order Approximation to the Optimum Checkpoint Interval.
ACM Comm. 17(9):530-531, September, 1974.

[Yourdon 72] E. Yourdon.
Reliability Measurements for Third Generation Systems.
In *Proceedings of the 1972 Annual Reliability and Maintainability Symposium*, pages 174-182. IEEE Comp. Soc., 1972.

[Ziegler 79] J.F. Ziegler and W.A. Lanford.
Effect of Cosmic Rays on Computer Memories.
Science 206:776-788, November, 1979.


ATE
LME